TUGAS AKHIR – IF184802

Analisis Kinerja Model Propagasi TwoRayGround pada Destination Sequence Distance Vector (DSDV) di lingkungan MANET

Muhammad Adnan Yusuf
NRP 05111340000001

Dosen Pembimbing I
Henning Titi Ciptaningtyas, S.Kom., M.Kom.

Dosen Pembimbing II
Dr. Eng. Radityo Anggoro, S. Kom., M. Sc.

Departemen Informatika
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember
Surabaya 2019

[Halaman ini sengaja dikosongkan]

TUGAS AKHIR – IF184802

Analisis Kinerja Model Propagasi TwoRayGround pada Destination Sequence Distance Vector (DSDV) di lingkungan MANET

Muhammad Adnan Yusuf
NRP 05111340000001

Dosen Pembimbing I
Henning Titi Ciptaningtyas, S.Kom., M.Kom.

Dosen Pembimbing II
Dr. Eng. Radityo Anggoro, S. Kom., M. Sc.

Departemen Informatika
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember
Surabaya 2019

[Halaman ini sengaja dikosongkan]

UNDERGRADUATE THESES – IF184802

Performance Analysis of TwoRayGround Propagation Model on Destination Sequence Distance Vector (DSDV) in MANET Environment

Muhammad Adnan Yusuf
NRP 05111340000001

Advisor I
Henning Titi Ciptaningtyas, S.Kom., M.Kom.

Advisor II
Dr. Eng. Radityo Anggoro, S. Kom., M. Sc.

DEPARTMENT OF INFORMATICS
Faculty of Information Technology and Communication
Institut Teknologi Sepuluh Nopember
Surabaya 2019

[Halaman ini sengaja dikosongkan]

LEMBAR PENGESAHAN

Analisis Kinerja Model Propagasi TwoRayGround pada Destination Sequence Distance Vector (DSDV) di lingkungan MANET

TUGAS AKHIR

Diajukan Guna Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Bidang Studi Arsitektur Jaringan Komputer
Program Studi S-1 Departemen Informatika
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember

Oleh :
Muhammad Adnan Yusuf
NRP : 05111340000001

Disetujui oleh Dosen Pembimbing Tugas Akhir :

Henning Titi Ciptaningtyas, S.Kom, M.Kom. ..... (pembimbing 1)
NIP: 198407082010122004

Dr.Eng. Radityo Anggoro, S.Kom., M.Sc. .....
NIP: 198410162008121002 (pembimbing 2)

SURABAYA
JANUARI 2019

[Halaman ini sengaja dikosongkan]

Analisis Kinerja Model Propagasi TwoRayGround pada Destination Sequence Distance Vector (DSDV) di lingkungan MANET

Nama Mahasiswa : Muhammad Adnan Yusuf
NRP : 05111340000001
Departemen : Informatika FTIK-ITS
Dosen Pembimbing 1 : Henning Titi Ciptaningtyas, S.Kom., M.Kom.
Dosen Pembimbing 2 : Dr. Eng. Radityo Anggoro, S. Kom., M. Sc.

ABSTRAK

MANET merupakan jaringan wireless yang terdiri dari kumpulan node yang bergerak yang memungkinkan untuk melakukan komunikasi secara langsung. Hal ini memungkinkan terjadinya komunikasi jaringan tanpa bergantung pada ketersediaan infrastruktur jaringan yang tetap.

Destination Sequence Distance Vector atau disingkat DSDV adalah salah satu metode routing yang menggunakan algoritma proactive. Algoritma proactive routing akan mengelola daftar tujuan dan rute terbaru masing-masing dengan cara mendistribusikan routing table ke seluruh jaringan, sehingga jalur lalu lintas(traffic) akan sering dilalui oleh routing table tersebut.

Pada Tugas Akhir ini, Implementasi pada lingkungan MANET dilakukan dengan menggunakan simulasi. Simulasi dilakukan dengan menggunakan aplikasi Network Simulator 2 (NS-2). Pada penelitian ini dilakukan analisa kinerja dari model propagasi TwoRayGround pada protokol routing DSDV di lingkungan MANET berdasarkan performa Packet Delivery Ratio(PDR), Routing Overhead(RO), dan End-to-End Delay.

Dari uji coba yang dilakukan, Model propagasi TwoRayGround pada protokol routing DSDV di lingkungan MANET memberikan rata-rata nilai Packet Delivery Ratio (PDR) senilai 33.15611% , rata-rata nilai Routing Overhead sebanyak 135.4064 paket ,dan rata-rata nilai End-to-End Delay senilai 0.20331 detik.

Kata kunci**: **MANET, DSDV, NS-2,TwoRayGround

Performance Analysis of TwoRayGround Propagation Model on Destination Sequence Distance Vector (DSDV) in MANET Environment

Student's Name : **Muhammad Adnan Yusuf**
Student's ID : **05111340000001**
Department : **Informatics FTIK-ITS**
Advisor 1 : **Henning Titi Ciptaningtyas, S.Kom., M.Kom.**
Advisor 2 : **Dr. Eng. Radityo Anggoro, S. Kom., M. Sc.**

ABSTRACT

MANET is a wireless network that consists of a set of moving nodes that allow it to connect directly. This allows communication networks to occur without relying on the availability of network infrastructure.

The Destination Sequence Distance Vector or DSDV is one of routing method that using proactive algorithm. Proactive routing algorithms will manage the destination list and the latest routing list for each of them by distributing the routing table on the entire network. As the result, The traffic lanes will often be traversed by the routing table.

In this Undergraduate Theses, The implementation on MANET enviroment is done using simulation. The simulation uses the Network Simulator 2 (ns-2) program. This research is done on the performance analysis of the TwoRayGround propagation model on the DSDV routing protocol in the MANET environment based on Packet Delivery Ratio (PDR), Routing Overhead (RO), and End to End delay.

From the experiments performed, The TwoRayGround propagation model on the DSDV routing protocol in the MANET environment gives the average value of the Packet Delivery Ratio (PDR) is 33.15611%, the average value of Routing Overhead (RO) is 135.4064 packets, and the average value of End-to-End Delay is 0.20331 seconds.

Keyword : MANET,DSDV,NS-2,TwoRayGround

KATA PENGANTAR

Alhamdulillahirabbil'alamin, segala puji bagi Allah SWT atas segala karunia dan rahmat-Nya penulis dapat menyelesaikan Tugas Akhir yang berjudul :

"Analisis Kinerja Model Propagasi TwoRayGround pada Destination Sequence Distance Vector (DSDV) di lingkungan MANET"

Harapan dari penulis semoga apa yang tertulis di dalam buku Tugas Akhir ini dapat bermanfaat bagi pengembangan ilmu pengetahuan saat ini, serta dapat memberikan kontribusi yang nyata.

Dalam pelaksanaan dan pembuatan Tugas Akhir ini tentunya sangat banyak bantuan yang penulis terima dari berbagai pihak, tanpa mengurangi rasa hormat penulis ingin mengucapkan terima kasih sebesar-besarnya kepada:

1. Allah SWT atas segala nikmat dan rizki yang telah di berikan kepada penulis.
2. Keluarga yang senantiasa memberikan dukungan penuh baik secara moril maupun materil.
3. Ibu Henning Titi Ciptaningtyas, S.Kom., M.Kom.selaku dosen pembimbing I yang telah bersedia meluangkan waktu untuk membimbing dan memotivasi penulis serta memberi arahan dalam mengerjakan Tugas Akhir.
4. Bapak Dr. Eng. Radityo Anggoro, S. Kom., M. Sc. selaku pembimbing II yang juga telah membantu, membimbing, dan memotivasi penulis dalam mengerjakan Tugas Akhir ini.
5. Bapak Dr.Eng. Darlis Herumurti, S.Kom, M.Sc. selaku kepala Departemen Informatika ITS.
6. Bapak/Ibu dosen, staf dan karyawan Jurusan Teknik Informatika ITS yang telah banyak memberikan dukungan, ilmu dan bimbingan yang tak ternilai harganya bagi penulis.

7. Teman-teman TC angkatan 2013 yang sudah bersama-sama jatuh bangun menjalani kuliah di kampus TC sejak maba hingga akhir kuliah.
8. Serta pihak-pihak lain yang tidak dapat disebutkan disini yang telah banyak membantu penulis dalam penyusunan Tugas Akhir ini.

Penulis telah berusaha sebaik-baiknya dalam menyusun Tugas Akhir ini, namun penulis mohon maaf apabila terdapat kekurangan, kesalahan maupun kelalaian yang telah penulis lakukan. Kritik dan saran yang membangun dapat disampaikan sebagai bahan perbaikan selanjutnya.

Surabaya, Januari 2019
Penulis

Muhammad Adnan Yusuf

DAFTAR ISI

DAFTAR GAMBAR

[Halaman ini sengaja dikosongkan]

DAFTAR TABEL

[Halaman ini sengaja dikosongkan]

DAFTAR KODE SUMBER

[Halaman ini sengaja dikosongkan]

BAB I
PENDAHULUAN

Pada bab ini akan dipaparkan mengenai garis besar Tugas Akhir yang meliputi latar belakang, tujuan, rumusan dan batasan permasalahan, metodologi pembuatan Tugas Akhir, dan sistematika penulisan.

1.1. Latar Belakang

Pada zaman era digital seperti sekarang ini dimana laju perkembangan teknologi sangat cepat, Tentu saja kebutuhan manusia akan komunikasi dan mengakses informasi pun semakin berkembang. Perangkat mobile seperti *smartphone*, *notebook* serta perangkat lainnya merupakan salah satu alasan kemajuan peradaban manusia, yang mana perangkat ini membuat sebuah koneksi berkomunikasi tanpa adanya jaringan infrastruktur yang tetap dan dapat dilakukan dengan posisi bergerak. Salah satu teknologi yang digunakan dalam membangun jaringan yang bersifat sementara dan bergerak adalah *Mobile Ad-hoc Network* (MANET).

MANET merupakan jaringan *wireless* yang terdiri dari kumpulan *node* yang bergerak yang memungkinkan untuk melakukan komunikasi secara langsung. Hal ini memungkinkan terjadinya komunikasi jaringan tanpa bergantung pada ketersediaan infrastruktur jaringan yang tetap. Sebagai contoh, di masa lalu untuk memancarkan paket data dalam jaringan, dibutuhkan infrastruktur seperti tower radio yang mahal dan cukup lama dalam pembangunannya namun, hal ini dapat di modernisasi dengan ketersediaan *router* sebagai *node* pada MANET yang memungkinkan jalur komunikasi yang lebih efektif dan efisien.

Hal penting yang harus di efektifkan dalam MANET adalah proses pengiriman paket data. Proses ini dikenal dengan nama *routing* yaitu proses pencarian rute tercepat untuk mengirimkan

data dari sumber ke tujuan. Terdapat beberapa metode routing pada jaringan MANET salah satunya *Destination Sequence Distance Vector* (DSDV).

Destination Sequence Distance Vector atau disingkat DSDV adalah salah satu metode *routing* yang menggunakan algoritma proaktif. Algoritma *proactive routing* akan mengelola daftar tujuan dan rute terbaru masing-masing dengan cara mendistribusikan *routing table* ke seluruh jaringan, sehingga jalur lalu lintas (*traffic*) akan sering dilalui oleh *routing table* tersebut.

Untuk mempelajari sistem dengan baik, implementasi pada lingkungan MANET dapat dilakukan dengan menggunakan simulasi. Simulasi dilakukan dengan menggunakan aplikasi *Network Simulator* 2 (NS-2). Pada penelitian ini akan dilakukan analisa kinerja dari model propagasi *TwoRayGround* pada protokol routing DSDV di lingkungan MANET berdasarkan *Packet Delivery Ratio*(PDR), *Routing Overhead*(RO), dan *End-to-End Delay*.

1.2. Rumusan Permasalahan

Rumusan masalah yang diangkat dalam Tugas Akhir ini dapat dipaparkan sebagai berikut:

1. Bagaimana kinerja protokol *routing* DSDV pada MANET?
2. Bagaimana hasil analisa studi kinerja model propagasi *TwoRayGround* pada protokol *routing* DSDV di lingkungan MANET?

1.3. Batasan Permasalahan

Beberapa batasan masalah yang terdapat dalam Tugas Akhir ini dapat dipaparkan sebagai berikut:

1. Protokol *routing* yang diuji coba adalah DSDV.
2. Model Propagasi yang akan dianalisa dalam Tugas Akhir ini adalah *TwoRayGround*.
3. Skenario uji coba dilakukan pada topologi MANET.

4. Simulasi pengujian protokol *routing* menggunakan *Network Simulator* 2.
5. Analisa kinerja didasarkan pada *Packet Delivery Ratio* (PDR), *Routing Overhead* (RO), dan *End-to-End Delay*.

1.4. Tujuan

Tujuan dari pengerjaan Tugas Akhir ini adalah menganalisa model propagasi *TwoRayGround* pada protokol *routing* DSDV di lingkungan MANET dengan menggunakan aplikasi *Network simulator* 2 (NS-2).

1.5. Manfaat

Tugas Akhir ini diharapkan dapat menghasilkan hasil analisa kinerja model propagasi *TwoRayGround* pada protokol DSDV yang efisien dengan parameter *Packet Delivery Ratio* (PDR), *Routing Overhead* (RO), dan *End-to-End Delay*.

1.6. Metodologi

Langkah-langkah yang ditempuh dalam pengerjaan Tugas Akhir ini yaitu:

1.6.1. Penyusunan proposal Tugas Akhir

Proposal tugas akhir ini berisi tentang deskripsi pendahuluan dari tugas akhir yang akan dibuat. Pendahuluan ini terdiriatas hal yang menjadi latar belakang diajukannya usulan tugas akhir, rumusan masalah yang diangkat, batasan masalah untuk Tugas Akhir, tujuan dari pembuatan Tugas Akhir, dan manfaat dari hasil pembuatan Tugas Akhir. Selain itu dijabarkan pula tinjauan pustaka yang digunakan sebagai referensi pendukung pembuatan tugas akhir. Sub-bab metodologi berisi penjelasan mengenai tahapan penyusunan Tugas Akhir mulai dari penyusunan proposal hingga penyusunan buku Tugas Akhir.

Terdapat pula sub-bab jadwal kegiatan yang menjelaskan jadwal pengerjaan Tugas Akhir.

1.6.2. Studi literatur

Studi literatur yang dilakukan dalam pengerjaan Tugas Akhir ini adalah mengenai peralatan dan metode yang digunakan. Literatur yang digunakan meliputi buku referensi, artikel, jurnal dan dokumentasi dari internet yang mendukung pengujian kinerja model propagasi *TwoRayGround* pada *Destination Sequence Distance Vector* (DSDV) di lingkungan MANET. Literatur-literatur yang dimaksud disebutkan sebagai berkut:

1. MANET
2. NS-2
3. DSDV
4. *TwoRayGround*

1.6.3. Implementasi

Tahap ini meliputi perancangan sistem bedasarkan studi literature dan pembelajaran konsep teknologi dan perangkat lunak yang ada. Tahap ini merupakan tahap yang paling penting dimana bentuk awal aplikasi yang akan diimplementasikan didefinisikan. Pada tahapan ini dibut prototype sistem, yang merupakan rancangan dasar dari sistem yang akan dibuat. Serta dilakukan desain suatu sistem dan desain proses-proses yang ada

1.6.4. Pengujian dan Evaluasi

Pada tahapan ini dilakukan uji coba terhadap sistem yang telah dibuat. Pengujian dan evaluasi dilakukan dengan mencari dan mengetahui nilai dari *Paket Delivery Ratio* (PDR), *Routing Overhead* (RO), dan *End-to-End Delay*. Tahap ini dilakukan untuk mengevaluasi jalannya sistem, mencari masalah yang mungkin timbul dan mengadakan perbaikan jika terdapat kesalahan.

1.6.5. Penyusunan Buku Tugas Akhir

Pada tahap ini dilakukan pendokumentasian dan pelaporan dari seluruh konsep, dasar teori, implementasi, proses yang telah dilakukan, dan hasil-hasil yang telah didapatkan selama pengerjaan Tugas Akhir.

1.7. Sistematika Penulisan

Buku Tugas Akhir ini bertujuan untuk mendapatkan gambaran dari pengerjaan Tugas Akhir ini. Selain itu, diharapkan dapat berguna untuk pembaca yang tertarik untuk melakukan pengembangan lebih lanjut. Secara garis besar, buku Tugas Akhir terdiri atas beberapa bagian seperti berikut ini.

Bab I Pendahuluan

Bab ini berisi latar belakang masalah, tujuan dan manfaat pembuatan Tugas Akhir, permasalahan, batasan masalah, metodologi yang digunakan, dan sistematika penyusunan Tugas Akhir.

Bab II Tinjauan Pustaka

Bab ini membahas beberapa teori penunjang yang berhubungan dengan pokok pembahasan dan mendasari pembuatan Tugas Akhir ini.

Bab III Analisis dan Perancangan Sistem

Bab ini membahas mengenai perancangan metode yang nantinya akan diimplementasikan dan dilakukan pengujian dari aplikasi yang dibangun.

Bab IV Implementasi

Bab ini berisi implementasi dari perancangan sistem yang sudah dilakukan pada tahap perancangan. Penjelasan berupa scenario *node node* pada jaringan *wireless*, konfigurasi sistem dan skrip analisis yang digunakan untuk menguji performa protocol *routing*

Bab V Pengujian dan Evaluasi

Bab ini membahas pengujian dari sistem dan performa dalam skenario mobilitas *ad hoc* yang dibuat dalam *network simulator*.

Bab VI Kesimpulan dan Saran

Bab ini berisi kesimpulan dari hasil pengujian yang dilakukan. Bab ini membahas saran-saran untuk pengembangan sistem lebih lanjut.

Daftar Pustaka

Merupakan daftar referensi yang digunakan untuk mengembangkan Tugas Akhir.

Lampiran

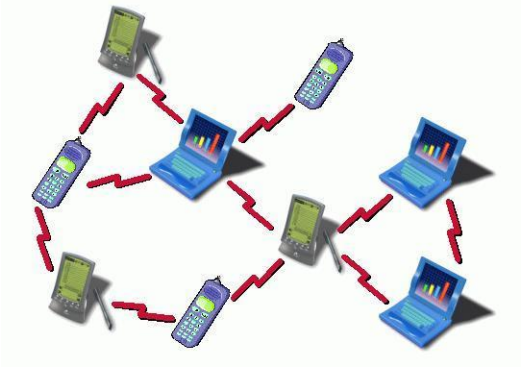
Merupakan bab tambahan yang berisi daftar istilah yang penting pada aplikasi ini.

BAB II
TINJAUAN PUSTAKA

Bab ini menjelaskan tentang tinjauan pustaka yang menjadi dasar pembuatan Tugas Akhir. Beberapa teori, pustaka, dan teknologi yang mendasari pengerjaan Tugas Akhir ini. Serta memberi gambaran terhadap alat, protokol *routing* serta definisi yang digunakan dalam pembuatan Tugas Akhir.

2.1. *Mobile Ad Hoc Network* (MANET)

Mobile Ad Hoc Network (MANET) merupakan sebuah jaringan yang terbentuk dari beberapa *node* yang bergerak bebas dan dinamis. MANET memungkinkan terjadinya komunikasi jaringan tanpa bergantung pada ketersediaan infrastruktur jaringan sebgai *host* dan *router*. Setiap *node* dapat saling melakukan komunikasi antara yang satu dengan yang lain tanpa adanya *access point*. [1]



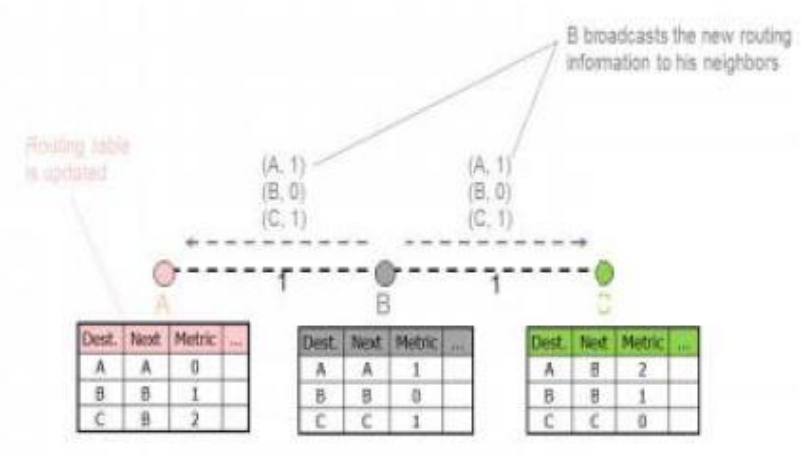
Gambar 2.1 Ilustrasi MANET [2]

Perangkat di jaringan MANET harus mampu mendeteksi keberadaan perangkat lain dan melakukan pengaturan yang diperlukan untuk melakukan komunikasi dan berbagi data. Pada MANET memungkinkan perangkat untuk mempertahankan koneksi ke jaringan serta dengan mudah menambahkan dan

menghapus perangkat pada jaringan. Karena pergerakan *node* yang dinamis, topologi jaringan dapat berubah dengan cepat dan tak terduga dari waktu ke waktu.

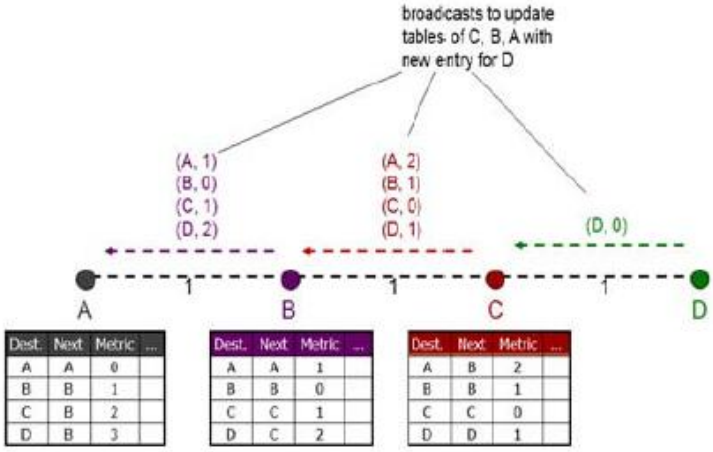
2.2. *Destination Sequence Distance Vector* (DSDV)

Destination Sequence Distance Vector (DSDV) adalah *routing protocol* yang bersifat proaktif dimana semua *mobile node* yang ada di jaringan memiliki rute yang valid dalam tabel *routing* ke tujuan yang ada di dalam jaringan. Dalam *routing* DSDV, Semua *mobile node* saling bertukar pesan *hello* untuk memberitahukan *node* mereka masing-masing. *Node* tetangga yang menerima paket *hello* akan menyimpan informasi *node* pengirim ke dalam tabel *routing* . Dengan cara ini, sebuah *node* akan mengetahui *node* tetangga. Kemudian Setiap *node* akan mengirimkan seluruh isi tabel routing ke *node* tetangga. Oleh karena itu, Setiap *node* akan memiliki jalur ke setiap *node* yang lain dan pada akhirnya paket *hello* akan dikirim oleh setiap *node* untuk memperbaruhi posisinya di dalam jaringan.



Gambar 2.2 Update *routing table* dengan pertukaran paket *Hello* [3]

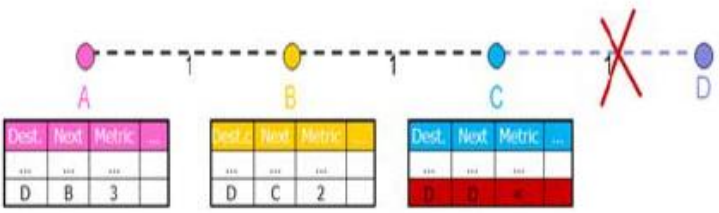
Pada Gambar 2.2, *node* A, B, dan C akan mengirim paket *hello*. *Node* B memiliki 2 *node* tetangga yakni *node* A dan C. Kemudian *node* C dan A akan memasukkan *node* B sebagai node tetangga mereka. *Node* B akan memasukkan *node* A dan C sebagai *node* tetangga. Kemudian pada tabel *routing* sebuah *node*, terdapat kolom *metric* dimana isinya berupa angka yang menunjukkan berapa *count of hops* ke sebuah node tetangga. Dengan itu, Setiap *node* dapat mengirimkan seluruh isi tabel *routing* ke *node* tetangga.



Gambar 2.3 Update routing table ketika ada sebuah *node* baru di jaringan [3]

Pada gambar 2.3, Ketika sebuah *no*de baru yakni *node* D tiba di jaringan, *Node* tersebut akan mengirimkan paket *hello*. *N*ode C akan menerima paket tersebut dan memasukkan *node* D sebagai *node* tetangga yang baru. Kemudian *node* C akan memberitahukan *node* tetangganya tentang *node* D yang baru. Kemudian *node B* akan menambahkan *node D* ke dalam table *routing* dan hal yang sama dilakukkan *node C* ke node A. Pada akhirnya semua node

yang ada menambahkan *node* D ke tabel *routing* mereka masing-masing.



Gambar 2.4 Update routing table ketika ada sebuah *node* yang terputus di jaringan [3]

Pada gambar 2.4, Terlihat *node* D bergerak menjauhi *node C* dan *node C* tidak dapat menjangkau *node* D. Sekarang *node* C harus memberitahukan ke seluruh jaringan bahwa *node D* tidak dapat terjangkau lagi dengan cara mengirimkan isi kolom *metric* yang berisi *node D* memiliki *infinity hops*. Seluruh *node* yang menerima pesan ini akan mengetahui bahwa *node D* sudah tidak ada melalui *node* C. [3]

2.3. Model Propagasi *TwoRayGround*

Model propagasi *TwoRayGround* merupakan model propagasi radio yang memperkirakan *path loss* antara antena pemancar dan antena penerima ketika berada pada LOS (*line of sight*). Sinyal penerima memiliki dua komponen, yaitu komponen LOS dan komponen *multipath* yang terbentuk oleh *single ground reflected wave*. Umumnya, kedua antena tersebut masing-masing memiliki ketinggian yang berbeda.

Jika dalam penggunaannya hanya tunggal, *single direct path* antara *base station* dan *mobile* terkadang hanya peralatan fisik biasa untuk propagasi dan rumus pada *free space* kurang akurat dalam *mobile radio channel*, Model propagasi *TwoRayGround*

merupakan model yang berguna karena berdasar pada optik geometri dan dapat digunakan untuk *direct path* dan refleksi dari *ground* antara *transmitter* dan *receiver*. Model ini diperkirakan sangat akurat untuk memperkirakan kekuatan sinyal dalam skala luas dengan jarak beberapa kilometer jika digunakan untuk sistem *mobile radio* dengan menggunakan menara yang tinggi. *Power* yang diterima dengan jarak d diberikan oleh persamaan (1) berikut :

$$\Pr(d) = \frac{PtGtGrht^2hr^2}{d^4L} \qquad (1)$$

Untuk penjelasan persamaan diatas, Pt adalah *power* yang dipropagasikan, ht dan hr adalah tinggi dari antena *transmitter* dan *receiver*, Gt dan Gr adalah tegangan dari antena *transmitter* dan *receiver*. nilai L diasumsikan sama dengan nilai L pada propagasi *free space*, L = 1. Untuk parameter yang lain, masih sama dengan parameter pada propagasi *free space*.

Persamaan tersebut menunjukkan semakin cepat *power* maka akan mengalami penurunan seiring peningkatan pada jarak. Namun demikian, model *TwoRayGround* tidak begitu memberikan hasil yang baik untuk jarak yang pendek dikarenakan osilasi yang disebabkan oleh kombinasi destruktif dan konstruktif dari *TwoRays* itu sendiri. Sebaliknya, pada model *free space* masih dapat digunakan asalkan nilai *d* diperkecil. [4]

2.4. *Network Simulator* 2 (NS-2)

NS-2 merupakan sebuah discrete event simulator yang di desain untuk membantu penelitian pada bidang jaringan komputer. Saat ini terdapat dua buah major version yang masih dikembangkan yaitu NS-2 dan NS-3. NS-3 dikembangkan sejak tanggal 1 juli 2006 dan sedang dikembangkan secara aktif

sedangkan pengembangan NS-2 tidak begitu aktif karena fokus pengembangan sebagian besar beralih ke NS-3. [5]

Versi terbaru NS-2 adalah NS-2.35 yang dirilis pada tanggal 4 november 2011. Dalam membuat sebuah simulasi jaringan komputer NS-2 menggunakan dua buah Bahasa pemograman, yaitu C++ dan Tcl. Bahasa C++ digunakan untuk mengimplemntasi bagian-bagian jaringan yang akan disimulasikan. Sedangkan Tcl digunakan untuk menulis scenario simulasi jaringan. NS-2 mendukung sistem operasi GNU/Lnux, FreeBSD, OS X dan Solaris. [6]

2.5. AWK

AWK adalah sebuah program *text processing* yang digunakan untuk mencari dan mengolah sebuah teks yang digunakan sebagai ekstrasi dari sebuah *dataset* . Nama AWK sendiri berasal dari penggabungan nama-nama lengkap sang pembuat yakni: Alfred V. Aho, Peter J. Weinberger, dan Brian W. Kernighan. AWK berisi kumpulan *command* (perintah) yang dijalankan pada *dataset* yang tersedia . AWK ditulis menggunakan Bahasa pemrogramannya sendiri yaitu awk *programming language*. [7]

Pada Tugas Akhir ini penulis menggunakan AWK untuk memproses hasil data *trace* yang dihasilkan dari simulasi menggunakan NS-2 untuk perhitungan *Packet Delivery Ratio* (PDR), *End-to-End Delay* (E2D), dan *Routing Overhead* (RO) dari hasil *trace* NS-2.

2.6. *Generator File Node Movement*

Sebuah universitas bernama Carnegie Mellon University (CMU) mengembangkan sebuah *tools* yang bernama *setdest* untuk menghasilkan *random movement* dari *nodes* dalam jaringan nirkabel. *Node movement* dihasilkan dengan kecepatan gerak maksimal yang spesifik menuju lokasi *node* acak ataupun lokasi *node* yang sudah ditentukan yang berada dalam kawasan yang telah

ditentukan. Apabila pergerakan *node* sudah sampai pada lokasi tujuan maka *node* tersebut akan berpindah ke lokasi selanjutnya. Pergerakan *node* tersebut dapat diatur untuk berhenti sementara selama waktu yang sudah ditentukan.

Sebelum menjalankan program simulasi pengguna harus menjalankan program 'setdest'. Format *command line* 'setdest' ditunjukan pada Kode Sumber 2.1 dan keterangannya ditunjukan pada tabel 2.1.

```
./setdest [-v version] [-n num_of_nodes] [-p pausetime] [-M maxspeed] [-t simtime] [-x maxx] [-y maxy] > [outdir/movement-file]
```

Kode Sumber 2.1 Format C*ommand Line* 'setdest'

Table 2.1 Penjelasan *Command Line* 'setdest'

Parameter	Keterangan
-v *version*	Menentukan versi 'Setdest' yang digunakan
-n *num_of_nodes*	Menentukan jumlah node yang dibuat dalam sebuah scenario
-p *pausetime*	Menentukan lama durasi berhenti pada sebuah paket apabila sudah sampai di tujuan
-M *maxspeed*	Menentukan kecepatan maksimum dari pengiriman paket.
-t *simtime*	Menentukan lama waktu jalannya simulasi.
-x *max* x	Menentukan panjang maksimum dari area simulasi.
-y *max* y	Menentukan lebar maksimum dari area simulasi

Command line 'setdest' yang sudah di-*generate* menghasilkan *file* yang berisi jumlah *node* dan pergerakan dari *node* yang sudah dibuat dalam *file* berformat .tcl. Selain pergerakan *node file* tersebut juga berisi tentang perpindahan rute. [8]

Untuk membuat skenario *node-movement* yang terdiri dari 100 *node*, bergerak dengan kecepatan maksimum 10 m/s, jeda rata-rata antar gerakan sebesar 3 detik, simulasi akan berhenti setelah 100 detik dengan batas topologi yang diartikan sebagai 500 x 500 meter, contoh *command line* seperti pada Kode Sumber 2.2.

```
./setdest -v 1 -n 100 -p 3 -M 10 -t 100 -x 500 -y 500 > scenario.txt
```

Kode Sumber 2.2 Contoh C*ommand Line* 'setdest'

2.7. *File Traffic Connection Pattern*

Random Traffic Connection pada TCP dan CBR dapat dikonfigurasi antar *mobilenodes* dengan skrip *traffic scenario generator*. Skrip ini dapat membantu kita untuk men-*generate* beban trafik yang berupa TCP atau CBR. Skrip yang digunakan untuk men-*generate* trafik adalah 'cbrgen.tcl'. Program 'cbrgen.tcl' disimpan di dalam direktori ~ns/indep-utils/cmu-scen-gen/. Program 'cbrgen.tcl' diigunakan sesuai dengan *command line* pada gambar 2.4 dan dengan keterangan yang diunjukan pada tabel 2.2. [9]

```
ns cbrgen.tcl [-type cbr|tcp] [-nn nodes] [-seed seed] [-mc connections] [-rate rate] > -traffic
```

Kode Sumber 2.3 Format *Command Line* 'cbrgen.tcl'

Table 2.2 Penjelasan *Command Line* 'cbrgen.tcl'

Parameter	Keterangan
-type cbr|tcp	Menentukan jenis *traffic* yang digunakan.
-nn *nodes*	Menentukan jumlah total *node*.
-s *seed*	Menentukan nilai *random seed*

-mc *connection*	Menentukan jumlah koneksi antar node.
-rate *rate*	Menentukan jumlah paket per detik yang terkirim.

Untuk membuat sebuah file koneksi CBR antar 50 *nodes*, yang memiliki maksimal 10 koneksi dengan nilai seed 1.0 dan jumlah paket per detik sebanyak 2.0 . Contoh *command line* seperti pada Kode Sumber 2.4.

```
ns cbrgen.tcl -type cbr -nn 50 -seed 1.0 -mc 10 -rate 2.0 > cbr.txt
```

Kode Sumber 2.4 Contoh C*ommand Line* 'cbrgen.tcl'

[Halaman ini sengaja dikosongkan]

BAB III
PERANCANGAN

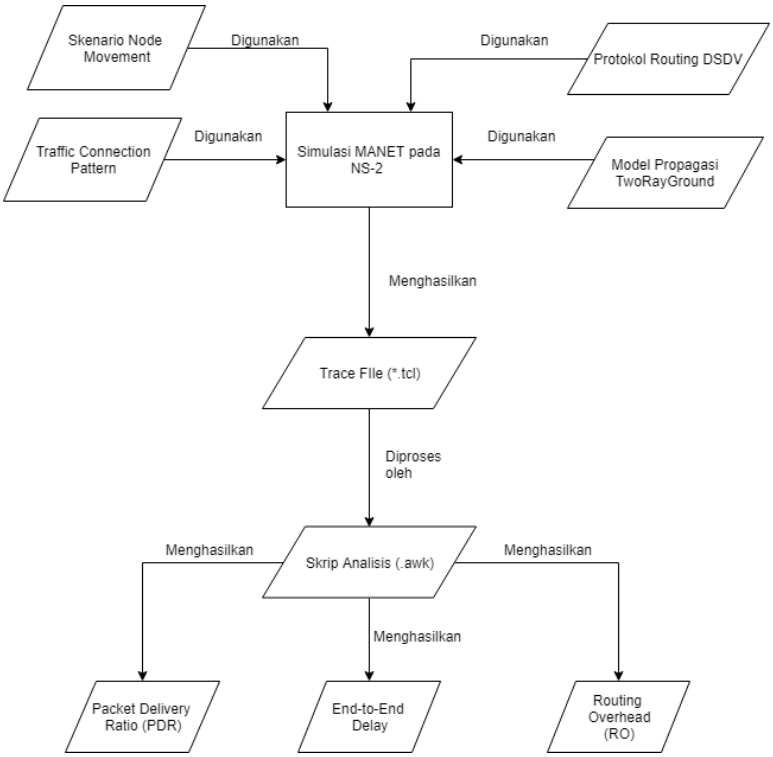
Pada bab ini akan dijelaskan mengenai dasar perancangan dari perangkat lunak yang akan dibangun dalam Tugas Akhir ini. Perancangan tersebut mencakup deskripsi umum aplikasi, arsitektur sistem, model fungsional dan diagram alur aplikasi.

3.1. Deskripsi Umum Sistem

Pada Tugas Akhir ini akan dilakukan analisis tentang performa model propagasi *TwoRayGround* dengan protokol *routing* DSDV pada Lingkungan MANET. Dalam pembuatan skenario penulis menggunakan *mobility generator* yang bersifat *random way point* dan telah ada pada *Network Simulator-2* (NS-2) yaitu dengan cara men-*generate file node movement* dan membuat koneksi antar *node* menggunakan *file traffic connection pattern*. Rancangan Simulasi yang dibuat dapat dilihat pada Gambar 3.1.

Dalam penelitian ini penulis akan menganalisa performa dari model Propagasi *TwoRayGround* pada simulasi skenario yang dijalankan pada NS-2 menggunakan *routing protocol* DSDV pada sistem operasi Linux.

Hasil proses uji coba dari tiap skenario akan menghasilkan sebuah *trace file* yang nantinya akan dilakukan analisis perhitungan metrik *packet delivery Ratio* (PDR), *End-to-End Delay*, dan *Routing Overhead* (RO). Dari hasil metrik tersebut dianalisis performa model propagasi *TwoRayGround* pada protokol *routing* DSDV.



Gambar 3.1 Diagram Rancangan Simulasi

3.2. Perancangan Skenario

Perancangan skenario uji coba dalam Tugas Akhir ini dibuat menjadi 3 skenario. Skenario pertama dibuat untuk melihat pergerakan *node* berdasarkan pada tiga kecepatan maksimal yaitu 5m/s, 10m/s, dan 15m/s. Skenario kedua dibuat untuk melihat pergerakan *node* berdasarkan pada tiga jumlah node yaitu 50 *node*, 75 *node*, dan 100 *node*. Skenario ketiga dibuat untuk melihat pergerakan *node* berdasarkan pada tiga radius transmisi yaitu 150m, 200m, dan 250m.

Semua skenario dimulai dengan menggunakan *mobility generation* kemudian penguji akan membuat koneksi dari skenario yang sudah ada menggunakan *file traffic connection* yang sudah ada pada NS-2. Sedangkan untuk koneksinya digunakan dua *node* untuk menentukan *node* pengirim dan *node* penerima paket.

3.2.1. Perancangan Skenario Node Movement (Mobility Generation)

Perancangan skenario dibuat dengan men-*generate file node movement* yang sudah disediakan oleh NS-2 dengan *tools* bernama 'setdest' yang akan digunakan dalam *file* tcl selama simulasi pada NS-2 sebagai bentuk pergerakan *node* yang berpindah-pindah.

Table 3.1 Penjelasan Skenario *node movement*

No.	Parameter	Spesifikasi
1	Jumlah *Node*	Skenario 1: 50 *node* Skenario 2: - 50 *node* - 75 *node* - 100 *node* Skenario 3: 50 *node*
2	Waktu Simulasi	100 detik
3	Area	800m x 800m
4	Kecepatan Maksimal	Skenario 1: 5 m/s, 10 m/s, 15 m/s Skenario 2: 10 m/s Skenario 3: 10 m/s
5	Sumber *Traffic*	CBR
6	Waktu Jeda (Dalam Detik)	10
7	Ukuran Paket	512 bytes
8	*Rate* Paket	0.25 paket per detik

9	Jumlah maksimal koneksi	1
10	Model mobilitas yang digunakan	*Random way point*

3.2.2. *Traffic-Connection Pattern*

Traffic-Connection dibuat dengan menjalankan program *cbrgren.tcl* yang telah ada pada NS-2 yang digunakan untuk memberikan koneksi dari *node-node* yang sudah dibuat dalam skenario diatas selama melakukan simulasi pada NS-2.

Table 3.2 Penjelasan *Traffic-connection pattern*

No.	Parameter	Spesifikasi
1	-type cbr|tcp	CBR
2	-nn *nodes*	2
3	-s *seed*	1.0
4	-mc *connection*	1
5	-rate *rate*	0.25

3.3. Perancangan Simulasi NS-2

Pada perancangan kode NS-2 dengan konfigurasi MANET. Dilakukan dengan menggunakan skenario mobilitas dan digabungkan dengan *script* TCL yang berisikan konfigurasi mengenai lingkungan simulasi. Konfigurasi lingkungan simulasi MANET pada NS-2 dapat dilihat di tabel 3.2

Table 3.3 Penjelasan simulasi NS-2

No	Parameter	Spesifikasi
1	*Network Simulator*	NS-2, 2.35
2	*Routing Protocol*	DSDV
3	Waktu Simulasi	100 detik
4	Area Simulasi	800m x 800m
5	Banyak *node*	Skenario 1: 50 *node* Skenario 2: 50 *node*, 75 *node*,100 *node*

		Skenario 3: 50
6	Radius Transmisi	Skenario 1:100 m Skenario 2:100 m Skenario 3:150 m, 200 m, 250 m
7	Agen Pengirim	*Constant Bit Rate* (CBR)
8	*Source/Destination*	Statis
9	*Packet Rate*	512 bytes
10	Ukuran Paket	32
11	Protokol MAC	IEEE 802.11
12	Propagasi Sinyal	*TwoRayGround*
13	Tipe Kanal	*Wireless Channel*

3.4. Perancangan Metriks Analisis

Berikut ini merupakan beberapa parameter yang dianalisis dalam Tugas Akhir ini:

3.4.1. *Packet Delivery Ratio* (PDR)

Packet delivery ratio merupakan perbandingan dari jumlah paket yang dikirimkan dengan paket yang diterima. *Packet delivery ratio* dihitung menggunakan persamaan 2, dimana *received* adalah jumlah paket data yang diterima dan *sent* adalah jumlah paket data yang dikirimkan. Semakin tinggi *packet delivery ratio* semakin berhasil pengiriman paket yang dilakukan.

$$Packet\ Delivery\ Ratio = \frac{\Sigma \text{received}}{\Sigma \text{sent}}\ x\ 100\% \quad (2)$$

3.4.2. *End-to-End Delay* (E2E)

End-to-end delay mengindikasikan *delay* transmisi paket dari *node* asal ke *node* tujuan. Total *delay* didapatkan dari akumulasi *delay-delay* kecil yang ada dalam jaringan. Nilai rata-rata *end to end delay* pada paket yang diterima bisa dihitung berdasarkan selisih waktu antara transmisi dan respon paket pada *Constant Bit Rate* (CBR) dan membaginya dengan jumlah total transmisi CBR menggunakan persamaan 3

$$End\ to\ End\ Delay = \sum_{i<sent}^{i=0} \frac{t^{received(i)} - t^{sent(i)}}{sent} \quad (3)$$

3.4.3. *Routing Overhead* (RO)

Routing overhead adalah jumlah paket routing yang ada didalam sebuah jaringan dibagi dengan jumlah keseluruhan paket data yang diterima, perhitungan menggunakan persamaan 4.

$$Routing\ Overhead = \sum_{i<sent}^{i=0} \frac{packet\ routing}{received} \quad (4)$$

BAB IV
IMPLEMENTASI

Bab ini akan menjelaskan tentang implementasi Tugas Akhir berdasarkan rancangan perangkat lunak yang dijelaskan meliputi lingkungan pembangunan perangkat lunak, implementasi skenario, implementasi simulasi NS-2, dan implementasi matiks analisis.

4.1. Lingkungan Implementasi Protokol

Sub bab ini menjelaskan tentang lingkungan implementasi Protokol yang dilakukan pada lingkungan:

Tabel 4.1 Spesifikasi Lingkungan Implementasi

Perangkat Keras	- Laptop Lenovo G40-80 o Prosesor Intel(R) Core(TM) i3-5005U CPU @ 2.00GHz o RAM 6 GB o SSD 240 GB
Perangkat Lunak	- Laptop Lenovo G40-80 o Sistem Operasi Ubuntu 16.04 o Network Simulator 2, 2.35

4.2. Implementasi Skenario

Implementasi skenario mobilitas MANET dimulai dengan melakukan pembuatan skenario oleh *Mobility Generation* untuk menempatkan dan menentukan *node-node* yang akan di simulasikan. Setelah membuat skenario maka penguji membuat jalur-jalur yang menghubungkan antar node dengan *traffic connection pattern*. implementasi skenarionya adalah sebagai berikut.

4.2.1. Skenario *File Node-Movement*(*Mobility Generation*)

Dalam implementasi yang dilakukan pada pembuatan skenario dengan *mobility generation* menggunakan *tools generate default* yang sudah disediakan oleh NS-2 yaitu 'setdest'. Skenario yang sudah di-*generate* ini akan digunakan untuk setiap simulasi yang dilakukan beradasarkan kecepatan maksimal yang berbeda beda. Algoritma yang digunakan untuk membuat skenario ini adalah *Random Way Point* sehinnga penempatan *node node* yang ada akan bersifat acak. . *Format command line* pada kode sumber 4.1 yang digunakan untuk menghasilkan gerakan acak pada node adalah sebagai berikut:

```
./setdest [-v version] [-n num_of_nodes] [-p
pausetime] [-M maxspeed] [-t simtime] [-x
maxx] [-y maxy] > [outdir/movement-file]
```

Kode Sumber 4.1 Format kode 'setdest'

Ketentuan-ketentuan yang diujicobakan pada skenario ini adalah versi 'setdest' simulator yaitu 1, jumlah node dalam skenario yaitu 50 untuk skenario 1 , 50, 75,dan 100 untuk skenario 2, dan 50 untuk skenario 3. Waktu jeda (*pause time*) yaitu 10 detik, kecepatan maksimalnya yaitu skenario 1 sebesar 5m/s,10m/s,dan 15m/s,skenario 2 sebesar 10m/s, dan skenario 3 sebesar 10m/s. Waktu simulasi yaitu 100 detik. Kemudian file mobilitas yang dihasilkan disimpan dalam direktori " ~ns/indep-utils/cmu-scen-gen/setdest/". Pada kode sumber 4.2 dapat dilihat contoh implementasi *command line* pada 'setdest' dengan berbagai kecepatan maksimal yang berbeda sesuai dan node sebanyak 50. Dan untuk setiap kecepatan maksimal tersebut dibuat 10 buah file untuk satu protokol routing dan satu model propagasi.

```
./setdest -v 1 -n 50 -p 10 -M 5 -t 100 -x 800
-y 800 > scenario.txt
./setdest -v 1 -n 50 -p 10 -M 10 -t 100 -x
800 -y 800 > scenario.txt
./setdest -v 1 -n 50 -p 10 -M 15 -t 100 -x
800 -y 800 > scenario.txt
```

Kode Sumber 4.2 Implementasi 'setdest'

4.2.2. File *traffic-connection pattern*

Dalam implementasi *random traffic connection* yang menggunakan tipe CBR ini dikonfigurasi dengan menggunakan skrip *traffic scenario generator*. Skrip *traffic scenario generator* akan menghasilkan file bernama 'cbrgen.tcl' yang akan digunakan unruk membuat jaringan atau hubungan antar *node-node* yang sudah dibuat pada skenario sebelumnya. ketika kita akan menggunakan file 'cbrgen.tcl' ini kita haris menentukan tipe koneksi yang digunakan(apakah CBR atau TCP), banyaknya *node* yang ada, koneksi maksimal yang diinginkan, dan nilai *random seed*. *Format command line* pada kode sumber 4.7 yang digunakan untuk menghasilkan gerakan acak pada *node* adalah sebagai berikut.

```
ns cbrgen.tcl [-type cbr|tcp] [-nn nodes] [-
seed seed] [-mc connections] [-rate rate]  >
-traffic
```

Kode Sumber 4.3 Format kode 'cbrgen.tcl'

Pada kode sumber 4.4 merupakan bentuk implementasi untuk menjalankan cbrgen.tcl untuk membuat file koneksi CBR diantara 2 *node* memiliki maksimal 1 koneksi dengan nilai *seed* 1.0 dan jumlah paket per detik sebanyak 0.25 yang disimpan dalam cbr.txt yang nantinya akan digunakan pada saat simulasi NS-2.

```
ns cbrgen.tcl -type cbr -nn 2 -seed 1.0 -mc
          1 -rate 0.25 > cbr.txt
```

Kode Sumber 4.4 Implementasi kode 'cbrgen.tcl'

4.3. Implementasi Simulasi pada NS-2

Implementasi simulasi NS-2 dilakukan dengan cara pendeskripsian lingkungan simulasi pada sebuah file dengan ekstensi *.tcl* dan .txt file-file ini berisi konfigurasi setiap *node* dan proses yang dilakukan selama simulasi berjalan.

Pada kode sumber 4.5 menunjukan skrip konfigurasi awal parameter paremeter yang diberikan untuk menjalakan simulasi MANET pada NS-2. Isi dari parameternya adalah sebegai berikut.

Table 4.1 Penjelasan simulasi NS-2

No.	Parameter	Spesifikasi
1	Channel/	Wireless Channel
2	Propagation/	TwoRayGround
3	Phy/ WirelessPhy	WirelessPhy
4	Mac/ 802.11	802.11
5	Queue Drotail/PriQueue	PriQueue
6	Antenna/OmniAntena	OmniAntena
7	Nilai X	800
8	Nilai Y	800
9	Nilai seed	0.0
10	Routing Protocol	DSDV
11	File traffic connection	"cbr.txt"
12	File node movement	"scenario.txt"

```
1.  set val(chan) Channel/WirelessChannel;
2.  set val(prop) Propagation/TwoRayGround;
3.  set val(netif)Phy/WirelessPhy;
4.  set val(mac)  Mac/802_11;
5.  set val(ifq)  Queue/DropTail/PriQueue;
```

```
6.   set val(ll)       LL;
7.   set val(ant)      Antenna/OmniAntenna;
8.   set val(ifqlen)   50;
9.   set val(nn)       50;
10.  set val(rp)       DSDV;
11.  set opt(x)        800;
12.  set opt(y)        800;
13.  set val(stop)     100;
14.  set val(seed)     0;
15.  set val(cp)       "cbr.txt";
16.  set val(sc)       "scenario.txt";
```

Kode Sumber 4.5 Konfigurasi parameter pada NS-2

Pada kode sumber 4.11 merupakan pengaturan dari *transmission range* yang digunakan pada simulasi. Nilai yang diubah ialah *RXThresh_ (receiver Sensitivity Threshold).* Nilai 1.42681e-08 pada variabel tersebut memiliki artian bahwa *range* atau jangkauan yang dapat dicapai ialah sejauh 100 meter.

```
Phy/WirelessPhy set RXThresh_ 1.42681e-08;
```

Kode Sumber 4.6 Pengaturan *Transmission range*

```
1. set ns_         [new Simulator]
2. set tracefd       [open trace.tr w]
3. set namtrace      [open simwrls.nam w]
4.
5. $ns_ trace-all $tracefd
6. $ns_ namtrace-all-wireless $namtrace
   $opt(x) $opt(y)
7.
8. # set up topography object
9. set topo        [new Topography]
```

```
10.  $topo load_flatgrid $opt(x) $opt(y)
11.
12.  set god_ [create-god $val(nn)]
13.
14.  ##
15. #  Create nn mobilenodes [$val(nn)] and
   attach them to the channel.
16. #
17.
18. # configure the nodes
19.          $ns_ node-config -adhocRouting
   $val(rp) \
20.               -llType $val(ll) \
21.               -macType $val(mac) \
22.               -ifqType $val(ifq) \
23.               -ifqLen $val(ifqlen) \
24.               -antType $val(ant) \
25.               -propType $val(prop) \
26.               -phyType $val(netif) \
27.               -channelType $val(chan) \
28.               -topoInstance $topo \
29.               -agentTrace ON \
30.               -routerTrace ON \
31.               -macTrace OFF \
32.               -movementTrace ON
33.
34.  for {set i 0} {$i < $val(nn) } { incr
   i } {
35.        set node_($i) [$ns_ node]
36.        $node_($i) random-motion 0;
37.  }
```

Kode Sumber 4.7 Pengaturan variabel global pada NS-2

Skrip yang ditunjukan pada kode sumber 4.7 merupakan skrip untuk pengaturan variabel global yang diawali dengan set ns merupakan kode untuk pembuatan simulator baru. Set *tracefd* dan *set namtrace* merupakan pengaturan untuk menentukan nama dari *trace file* dan *file network* yang akan dihasilkan dan disimpan setelah simulasi selesai dilaksanakan. *Set topo* merupakan pengaturan untuk objek topografi berdasarkan pada luas koordinat yang telah dikonfigurasi sebelumnya. *Set god* dan *node config –channelType* merupakan konfigurasi yang dilakukan pada *node-node* yang akan dibuat. Terakhir dilakukan perulangan untuk membuat pergerakan dari *node-node*. *Node-node* yang dibuat tidak dapat melakukan pergerakan secara acak karena pergerakan *node* merupakan *trace file* yang dihasilkan oleh *mobility generator*.

Skrip pada kode sumber 4.8 merupakan bagian akhir dari keseluruhan skrip yang digunakan untuk menginisiasi penempatan awal *node-node* yang dibuat pada skenario *node-movement* (*mobility generation*), pergerakan *node* tersebut selama waktu simulasi dilakukan dan melakukan konfigurasi pengiriman paket data yang dilakukan nantinya dihasilkan pada *file output .tr* pada potongan skrip tersebut, akan dipanggil file skenario *node-movement* (*mobility generation*) dan *traffic-connection pattern* kemudian pengiriman paket data dimulai pada detik ke-0 dan diberhentikan pada detik ke 100 seperti yang telah di konfigurasi sebelumnya.

```
# Define node movement model
puts "Loading Conneciton Pattern ...."
source $val(cp)

#Define traffic mode
puts "Loading scenarion file...."
source $val(sc)
```

```
# Define node initial position in nam
for {set i 0} {$i < $val(nn)} { incr i } {

# 30 defines the node size for nam
$ns_ initial_node_pos $node_($i) 30
}

# Telling nodes when the simulation ends
for {set i 0} {$i < $val(nn) } { incr i } {
    $ns_ at $val(stop).0 "$node_($i)
reset";
}

$ns_ at $val(stop).0002 "puts \"NS
EXITING....\";
$ns_ halt"

puts $tracefd "M 0.0 nn $val(nn) x $opt(x)
y $opt(y) rp $val(rp)"
puts $tracefd "M 0.0 sc $val(sc) cp
$val(cp) seed $val(seed)"
puts $tracefd "M 0.0 prop $val(prop) ant
$val(ant)"

puts "Starting Simulation...."
$ns_ run
```

Kode Sumber 4.8 Menginisiasi penempatan awal node

4.4. *Implementasi Metriks Analisis*

Setelah selesai melakukan simulasi dengan NS-2 maka akan tercipta sebuah *trace file* yang berisi nilai nilai yang dapat kita gunakan untuk melakukan analisis dari simulasi yang sudah dilaksanakan. Dalam Tugas Akhir ini penulis akan melakukan analisis dari nilai *Packet Delivery Ratio* (PDR), *Routing Overhead* (RO), dan *End to End Delay* (E2E).

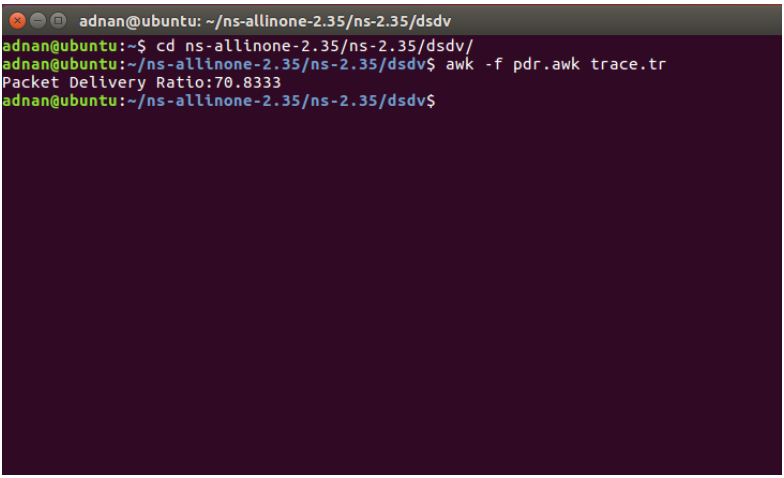
4.4.1. Implementasi *Packet Delivery Ratio*

Packet Delivery Ratio didapatkan dengan membandingkan pengiriman dan penerimaan data yang dikirimkan oleh agen. Pada Tugas Akhir ini, penulis menggunakan agen dengan pengiriman data CBR. Kemudian, pada Persamaan 2 telah dijelaskan rumus untuk menghitung *packet delivery ratio*. Skrip awk untuk menghitung *packet delivery ratio* berdasarkan kedua informasi tersebut dapat dilihat pada kode sumber 7.6. Cara menjalankan skrip awk dapat dilihat pada perintah awk di bawah ini.

```
awk -f pdr.awk trace.tr
```

Kode Sumber 4.9 Menjalankan file pdr.awk

Hasil dari perintah yang dijalankan adalah *packet delivery ratio* dari simulasi yang telah dijalankan dapat dilihat pada gambar 4.1.



Gambar 4.1 Contoh hasil nilai PDR

4.4.2. Implementasi *End-to-End Delay*

Perhitungan *end to end delay* didasarkan pada Persamaan 3 dan sudah dijelaskan pada bagian 3.4.2 Skrip awk untuk menghitung *end to end delay* berdasarkan kedua informasi tersebut dapat dilihat pada kode sumber 7.7. Contoh perintah untuk memanggil skrip tcl untuk menganilisis *trace file* dan *outputnya* dapat dilihat pada perintah awk di bawah ini.

```
awk -f e2e.awk trace.tr
```

Kode Sumber 4.10 Menjalankan file e2e.awk

Hasil dari perintah yang dijalankan adalah *end-to-end delay* dari simulasi yang telah dijalankan dapat dilihat pada gambar 4.2.

Gambar 4.2 Contoh hasil nilai E2E

4.4.3. Implementasi *Routing Overhead*

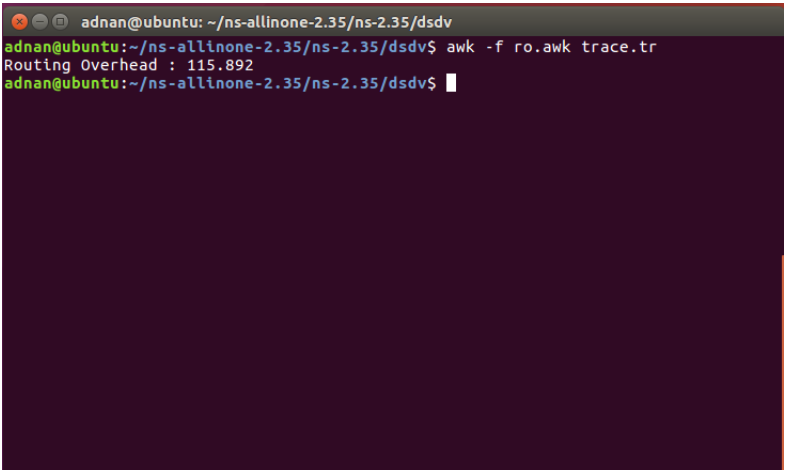
Perhitungan *routing overhead* didasarkan pada Persamaan 4 dan sudah dijelaskan pada bagian 3.4.3 Skrip awk untuk menghitung *routing overhead* berdasarkan kedua informasi tersebut dapat dilihat pada kode sumber 7.5.

Contoh perintah untuk memanggil skrip tcl untuk menganilisis *trace file* dan *outputnya* dapat dilihat pada perintah awk di bawah ini.

```
awk -f ro.awk trace.tr
```

Kode Sumber 4.11 Menjalankan file ro.awk

Hasil dari perintah yang dijalankan adalah *end-to-end delay* dari simulasi yang telah dijalankan dapat dilihat pada Gambar 4.12.



Gambar 4.3 Contoh hasil nilai RO

[Halaman ini sengaja dikosongkan]

BAB V
UJI COBA DAN EVALUASI

Bab ini membahas pengujian dan evaluasi pada dari skenario NS-2 yang telah dibuat. Pengujian fungsionalitas akan dibagi ke dalam beberapa skenario pengujian.

5.1. Lingkungan Pengujian

Lingkungan pengujian sistem pada pengerjaan Tugas Akhir ini dilakukan pada lingkungan dan alat kakas seperti yang tertera pada Tabel 5..

Tabel 5.1 Lingkungan Pengujian Sistem

Komponen	**Spesifikasi**
CPU	Intel(R) Core(TM) i3-5005U @ 2.00 GHz
Sistem Operasi	Ubuntu 16.04 LTS
Linux Kernel	Linux kernel 4.4
Memori	8 GB
Penyimpanan	SSD 240 GB

5.2. Kriteria Pengujian

Pengujian pada skenario yang dihasilkan oleh *mobility generator* default dari NS-2 menggunakan beberapa kriteria. Pada tabel 5.2 menunjukan kriteria-kriteria yang ditentukan didalam skenario.

Table 5.2 Penjelasan skenario pengujian

Kriteria	Spesifikasi
Skenario	MANET
Kecepatan Maksimal Perpindahan node (m/s)	Skenario 1 : 5m/s, 10m/s, 15m/s Skenario 2 : 10m/s

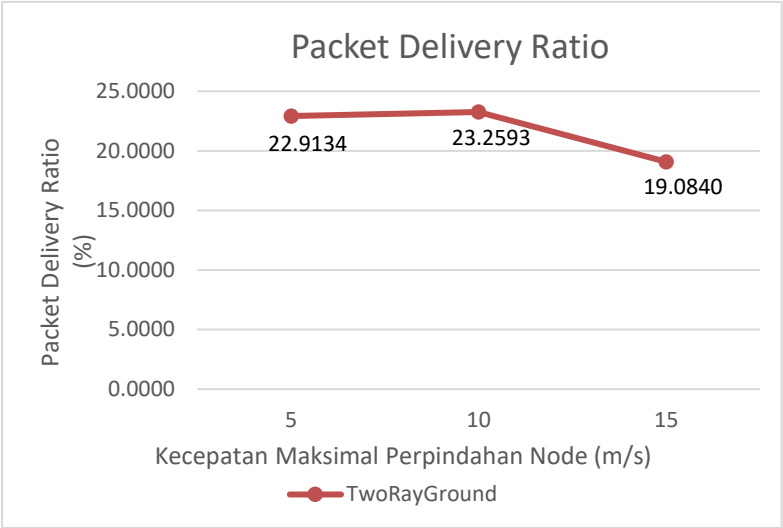
	Skenario 3 : 10m/s
Jumlah *node*	Skenario 1 : 50 *node* Skenario 2 : 50 *node* ,75 *node*, 100 *node*
Radius Transmisi	Skenario 1 : 100 m Skenario 2 : 100 m Skenario 3 : 150 m, 200 m, 250 m
Jumlah percobaan	10
Jarak *node* 1 dan *node* 2	Acak
Posisi awal *node*	Acak
Pergerakan	Acak
Protokol Routing	DSDV
Pengiriman Paket Data	0 – 100 Detik

5.3. Analisis *Packet Delivery Ratio (PDR)*

Dari hasil analisis yang dilakukan setelah melakukan uji coba berulang kali, Penulis mendapatkan nilai rataan pada *packet delivery ratio* dengan menggunakan model propagasi *TwoRayGround*.

Table 5.3 Nilai hasil PDR pada Skenario 1

Kecepatan Maksimal Perpindahan *Node* (m/s)	PDR (%)
5	22.9134
10	23.2593
15	19.0840



Gambar 5.1 Grafik nilai PDR pada Skenario 1

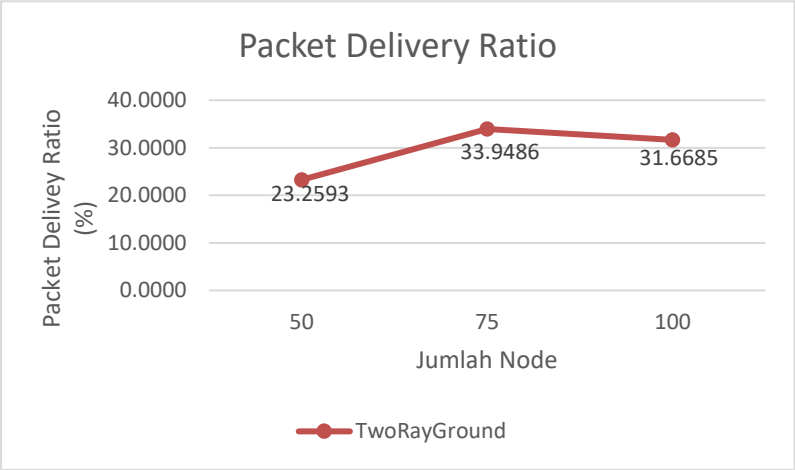
Pada tabel 5.3 dan gambar 5.1 menunjukan performa PDR yang dihasilkan oleh model propagasi *TwoRayGround* pada jaringan MANET dengan menggunakan skenario *node-Movement* (*mobility generation*) yang bersifat *Random Way Point*.

Bisa dilihat bahwa nilai rata rata dari *Packet Delivery Ratio* menunjukan nilai yang bersifat fluktuatif. Bisa dilihat bahwa nilai rata rata pada kecepatan maksimal 5m/s adalah 22.9134% sementara pada kecepatan maksimal 10m/s memiliki nilai 23.2593% dan pada kecepatan maksimal 15m/s bernilai 19.0840%. Hal ini dapat terjadi karena pada model transmisi TwoRayGround, skenario yang yang dihasilkan oleh file node-movement (mobility generation) tidak mampu untuk mengidentifikasi/memperhatikan lingkungan sekitar area simulasi, pengaruh ketinggian antena, dan pengaruh radius transmisi pada node yang menyebabkan pengiriman paket antar node yang lebih cepat dan dinamis. pergerakan *node* yang lebih dinamis inilah yang memungkinkan terjadinya banyak rute putus saat pengiriman paket data ataupun

kegagalan pembentukan tabel routing yang dibuat oleh DSDV sehingga menyebabkan paket data yang dikirimkan tidak sampai ke tujuan. Hal lain yang dapat mempengaruhi penurunan ataupun peningkatan nilai PDR yang dihasilkan adalah penempatan dan pergerakan secara acak yang di implementasikan pada skenario yang dihasilkan oleh file *node-movement* (*mobility generation*).

Table 5.4 Nilai hasil PDR pada Skenario 2

Jumlah *Node*	PDR (%)
50	23.2593
75	33.9486
100	31.6685



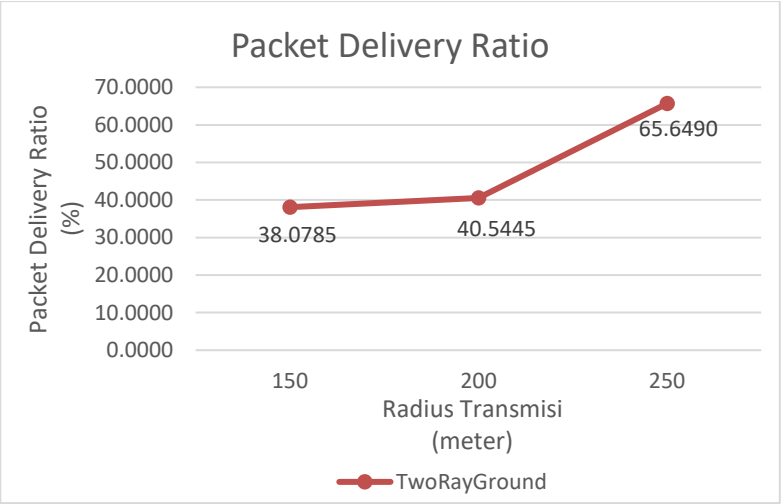
Gambar 5.2 Grafik nilai PDR pada Skenario 2

Pada tabel 5.4 dan gambar 5.2 menunjukan performa PDR yang dihasilkan oleh model propagasi *TwoRayGround* pada jaringan MANET dengan menggunakan skenario *node-Movement* (*mobility generation*) yang bersifat *Random Way Point*.

Bisa dilihat bahwa nilai rata rata dari *Packet Delivery Ratio* menunjukan nilai yang bersifat fluktuatif. Bisa dilihat bahwa nilai rata rata pada 50 *node* adalah 23.2593 % sementara pada 75 *node* meningkat sebanyak 10.6893 % sehingga memiliki nilai 33.9486 % dan pada 100 *node* mengalami penurunan sebanyak 2.2801 % sehingga memiliki nilai 31.6685 %. Hal ini dapat terjadi karena pada model transmisi TwoRayGround, skenario yang yang dihasilkan oleh file node-movement (mobility generation) tidak mampu untuk mengidentifikasi/memperhatikan lingkungan sekitar area simulasi pengaruh ketinggian antena, dan pengaruh radius transmisi pada node yang menyebabkan pengiriman paket antar node yang lebih cepat dan dinamis. pergerakan *node* yang lebih dinamis inilah yang memungkinkan terjadinya banyak rute putus saat pengiriman paket data ataupun kegagalan pembentukan tabel routing yang dibuat oleh DSDV sehingga menyebabkan paket data yang dikirimkan tidak sampai ke tujuan. Hal lain yang dapat mempengaruhi penurunan ataupun peningkatan nilai PDR yang dihasilkan adalah penempatan dan pergerakan secara acak yang di implementasikan pada skenario yang dihasilkan oleh file *node-movement* (*mobility generation*).

Table 5.5 Nilai hasil PDR pada Skenario 3

Radius Transmisi (meter)	PDR (%)
150	38.0785
200	40.5445
250	65.6490



Gambar 5.3 Grafik nilai PDR pada Skenario 2

Pada tabel 5.5 dan gambar 5.3 menunjukan performa PDR yang dihasilkan oleh model propagasi *TwoRayGround* pada jaringan MANET dengan menggunakan skenario *node-Movement* (*mobility generation*) yang bersifat *Random Way Point.*

Bisa dilihat bahwa nilai rata rata dari *Packet Delivery Ratio* menunjukan nilai yang bersifat cenderung meningkat. Bisa dilihat bahwa nilai rata rata pada radius transmisi 150 m adalah 38.0785 % sementara pada radius transmisi 200 m meningkat sebanyak 2.466 % sehingga memiliki nilai 40.5445 % dan pada 100 *node* mengalami peningkatan sebanyak 25.1045 % sehingga menghasilkan nilai sebesar 65.6490%. Hal ini dapat terjadi karena pada model transmisi TwoRayGround, skenario yang yang dihasilkan oleh file node-movement (mobility generation) tidak mampu untuk mengidentifikasi/memperhatikan lingkungan sekitar area simulasi pengaruh ketinggian antena, dan pengaruh radius transmisi pada node yang menyebabkan pengiriman paket antar node yang lebih cepat dan dinamis. pergerakan *node* yang lebih dinamis inilah yang memungkinkan terjadinya banyak rute putus

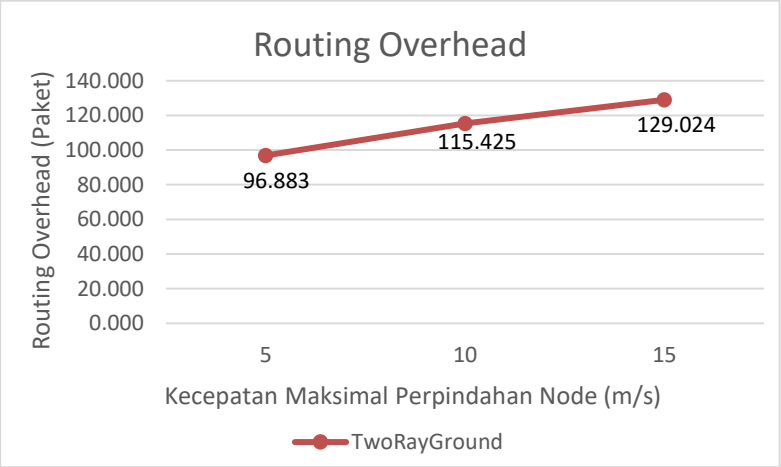
saat pengiriman paket data ataupun kegagalan pembentukan tabel routing yang dibuat oleh DSDV sehingga menyebabkan paket data yang dikirimkan tidak sampai ke tujuan. Hal lain yang dapat mempengaruhi penurunan ataupun peningkatan nilai PDR yang dihasilkan adalah penempatan dan pergerakan secara acak yang di implementasikan pada skenario yang dihasilkan oleh file *node-movement* (*mobility generation*).

5.4. Analisis *Routing Overhead(RO)*

Dari hasil analisis yang dilakukan setelah melakukan uji coba berulang kali, Penulis mendapatkan nilai rataan pada *Routing Overhead* dengan menggunakan model propagasi *TwoRayGround*.

Table 5.6 Nilai hasil RO pada Skenario 1

Kecepatan Maksimal Perpindahan *Node* (m/s)	RO (paket)
5	96.883
10	115.425
15	129.024

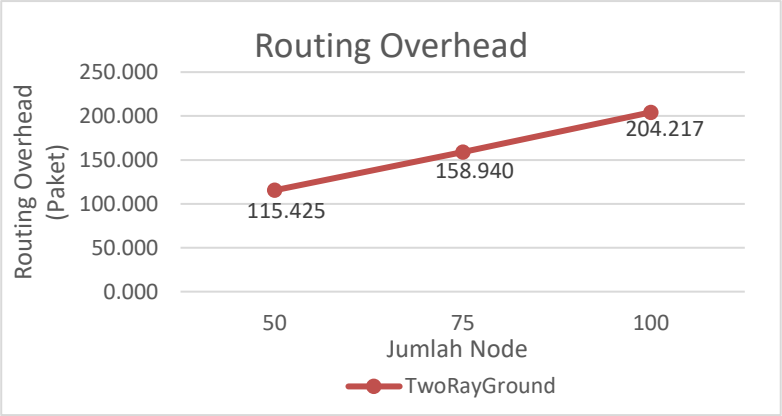


Gambar 5.4 Grafik nilai RO pada Skenario 1

Dari hasil skenario yang sudah dibuat bisa dilihat bahwa nilai rata rata dari *Routing Overhead* yang dihasilkan menunjukan nilai yang relatif meningkat. Pada kecepatan maksimal 5m/s adalah 96.883 paket sementara pada kecepatan maksimal 10m/s adalah 115.425 paket dan pada kecepatan maksimal 15m/s adalah 129.024 paket. Hal ini terjadi karena ketika node bergerak cepat, pengiriman paket routing berjenis send maupun forward yang dihasilkan lebih banyak. Dengan semakin banyaknya paket routing send dan forward yang dihasilkan, pengiriman paket data yang dilakukan memiliki kemungkinan yang lebih besar untuk sampai ke node tujuan.

Table 5.7 Nilai hasil RO pada Skenario 2

Jumlah *Node*	RO (paket)
50	115.425
75	158.940
100	204.217

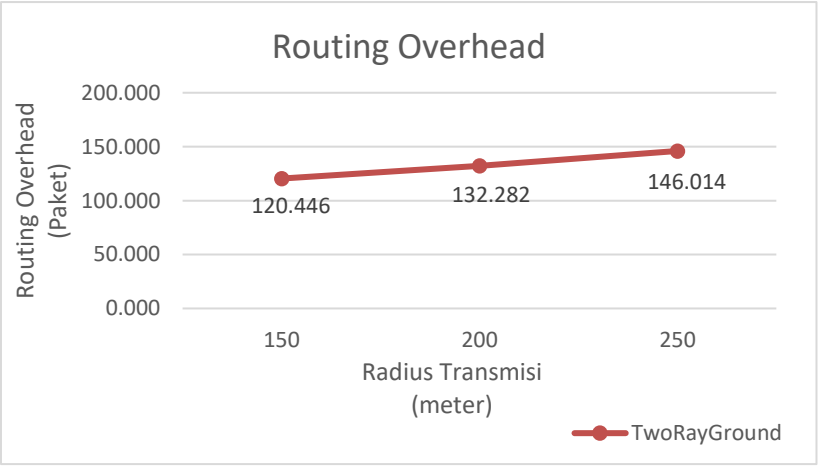


Gambar 5.5 Grafik nilai RO pada Skenario 2

Dari hasil skenario yang sudah dibuat bisa dilihat bahwa nilai rata rata dari *Routing Overhead* yang dihasilkan menunjukan nilai yang relatif meningkat. Pada jumlah node 50 *node* adalah 115.425 paket sementara pada jumlah node 75 *node* adalah 158.940 paket dan pada pada jumlah node 100 *node* adalah 204.217 paket. Hal ini terjadi karena ketika jumlah *node* bertambah banyak, pengiriman paket routing berjenis send maupun forward yang dihasilkan lebih banyak. Dengan semakin banyaknya paket routing send dan forward yang dihasilkan, pengiriman paket data yang dilakukan memiliki kemungkinan yang lebih besar untuk sampai ke node tujuan.

Table 5.8 Nilai hasil RO pada Skenario 3

Radius Transmisi (meter)	RO (paket)
150	120.446
200	132.282
250	146.014



Gambar 5.6 Grafik nilai RO pada Skenario 3

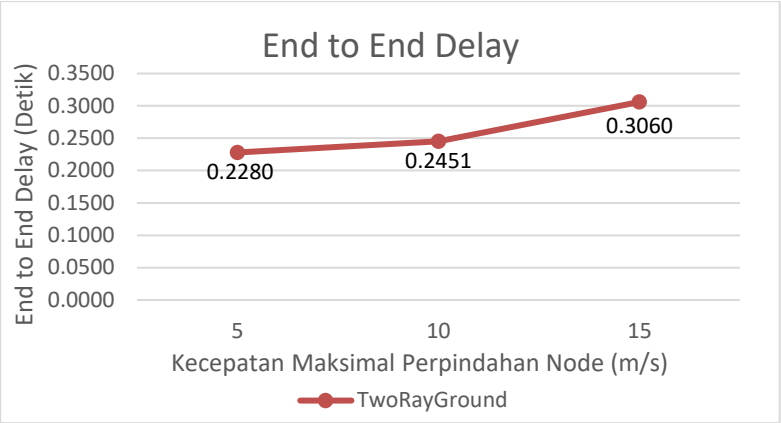
Dari hasil skenario yang sudah dibuat bisa dilihat bahwa nilai rata rata dari *Routing Overhead* yang dihasilkan menunjukan nilai yang relatif meningkat. Pada radius transmisi 150m adalah 120.446 paket sementara pada radius transmisi 200m adalah 132.282 paket dan pada radius transmisi 250m adalah 146.014 paket. Hal ini terjadi karena ketika radius transmisi semakin jauh, pengiriman paket routing berjenis send maupun forward yang dihasilkan lebih banyak. Dengan semakin banyaknya paket routing send dan forward yang dihasilkan, pengiriman paket data yang dilakukan memiliki kemungkinan yang lebih besar untuk sampai ke node tujuan.

5.5. Analisis *End-to-End Delay (E2E)*

Dari hasil analisis yang dilakukan setelah melakukan uji coba berulang kali, Penulis mendapatkan nilai rataan pada *End-to-End Delay* dengan menggunakan model propagasi *TwoRayGround*.

Table 5.9 Hasil nilai E2E pada Skenario 1

Kecepatan Maksimal Perpindahan *Node* (m/s)	*End to End delay* (Detik)
5	0.2280
10	0.2451
15	0.3060

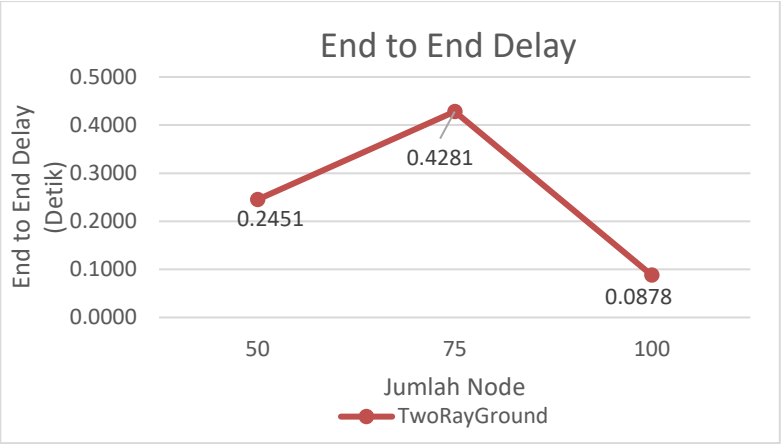


Gambar 5.7 Grafik hasil nilai E2E pada Skenario 1

Pada tabel 5.9 dan gambar 5.7 menunjukan pengujian model propagasi *TwoRayGround* untuk nilai rata rata dari *End to End Delay* relatif meningkat pada jumlah waktu yang dibutuhkan untuk mengirimkan paket data yang dikirimkan. Pada kecepatan maksimal 5m/s rata rata nilainya adalah 0.2280 detik sementara pada kecepatan maksimal 10m/s adalah 0.2451 detik dan pada kecepatan maksimal 15m/s adalah 0.3060 detik. Hal ini dapat terjadi akibat mobilitas pengiriman paket antar *node* yang sangat dinamis pada skenario *node-movement* (*mobility generation*) yang dihasilkan oleh model propagasi *TwoRayGround*. Mobilitas yang sangat dinamis ini memungkinkan terjadinya rute putus saat pengiriman paket data sehinga pengiriman paket dimasukkan ke dalam antrian dan menunggu rute baru terbentuk sebelum pengiriman paket data dilanjutkan kembali.

Table 5.10 Hasil nilai E2E pada Skenario 2

Jumlah *Node*	*End to End delay* (Detik)
50	0.2451
75	0.4281
100	0.0878

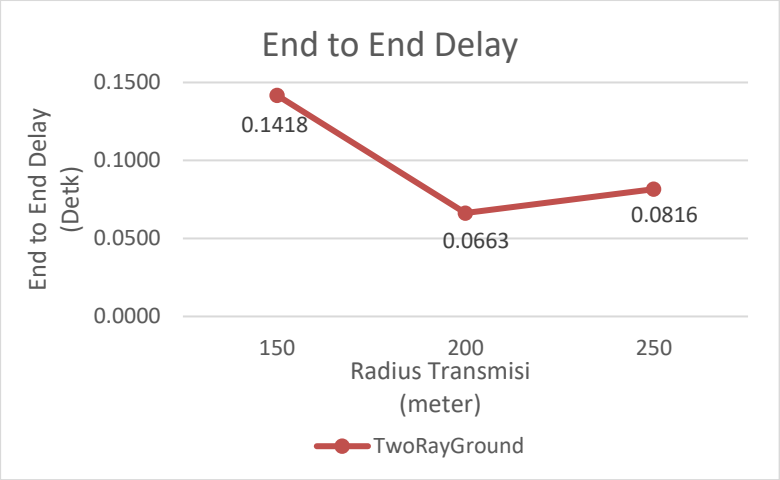


Gambar 5.8 Grafik hasil nilai E2E pada Skenario 2

Pada tabel 5.10 dan gambar 5.8 menunjukan pengujian model propagasi *TwoRayGround* untuk nilai rata rata dari *End to End Delay* relatif fluktuatif pada jumlah waktu yang dibutuhkan untuk mengirimkan paket data yang dikirimkan. Pada jumlah *node 50 node* rata rata nilainya adalah 0.2451 detik sementara pada jumlah *node* 75 *node* adalah 0.4281 detik dan pada jumlah *node* 100 *node* adalah 0.0878 detik. Hal ini dapat terjadi akibat mobilitas pengiriman paket antar *node* yang sangat dinamis pada skenario *node-movement* (*mobility generation*) yang dihasilkan oleh model propagasi *TwoRayGround*. Mobilitas yang sangat dinamis ini memungkinkan terjadinya rute putus saat pengiriman paket data sehinga pengiriman paket dimasukkan ke dalam antrian dan menunggu rute baru terbentuk sebelum pengiriman paket data dilanjutkan kembali.

Table 5.11 Hasil nilai E2E pada Skenario 3

Radius Transmisi (meter)	*End to End delay* (Detik)
150	0.1418
200	0.0663
250	0.0816



Gambar 5.9 Grafik hasil nilai E2E pada Skenario 3

Pada tabel 5.11 dan gambar 5.9 menunjukan pengujian model propagasi *TwoRayGround* untuk nilai rata rata dari *End to End Delay* relatif fluktuatif pada jumlah waktu yang dibutuhkan untuk mengirimkan paket data yang dikirimkan. Pada radius transmisi 150m rata rata nilainya adalah 0.1418 detik sementara pada radius transmisi 200m adalah 0.0663 detik dan pada radius transmisi 200m adalah 0.0816 detik. Hal ini dapat terjadi akibat mobilitas pengiriman paket antar *node* yang sangat dinamis pada skenario *node-movement* (*mobility generation*) yang dihasilkan oleh model propagasi *TwoRayGround*. Mobilitas yang sangat dinamis ini memungkinkan terjadinya rute putus saat pengiriman paket data sehinga pengiriman paket dimasukkan ke dalam antrian dan menunggu rute baru terbentuk sebelum pengiriman paket data dilanjutkan kembali.

[Halaman ini sengaja dikosongkan]

BAB VI
KESIMPULAN DAN SARAN

Pada bab ini akan diberikan kesimpulan yang diambil selama pengerjaan Tugas Akhir serta saran-saran tentang pengembangan yang dapat dilakukan terhadap Tugas Akhir ini di masa yang akan datang.

6.1. Kesimpulan

Kesimpulan yang dapat diambil dalam Tugas Akhir ini adalah sebagai berikut:

1. Berikut ini merupakan nilai rataan dari PDR, RO,E2E pada MANET menggunakan protokol routing DSDV.
    - Nilai rataan PDR yang didapat adalah 33.15611%.
    - Nilai rataan RO yang didapat adalah 135.4064 paket yang terkirim.
    - Nilai rataan E2E yang didapat adalah 0.20331 detik untuk mengirimkan paket.
2. Skenario Manet yang dihasilkan oleh *node-movement*(*mobility generation*) dan dijalankan menggunakan model propagasi *TwoRayGround* dengan penambahan kecepatan maksimal perpindahan *node* memiliki performa sebagai berikut.
    - Performa *Packet Delivery Ratio* yang dihasilkan fluktuatif mulai dari 22.9134% pada kecepatan maksimal 5m/s menjadi 23.2593% pada kecepatan maksimal 10m/s dan menjadi 19.0840% pada kecepatan maksimal 15m/s.
    - Performa *Routing Overhead* yang dihasilkan memiliki nilai semakin meningkat mulai dari 96.883 paket pada kecepatan maksimal 5m/s menjadi 115.425 paket pada kecepatan maksimal 10m/s dan berubah menjadi 129.024 paket pada kecepatan maksimal 15m/s.

- Performa *End to End Delay* yang dihasilkan memiliki nilai *delay* yang meningkat mulai dari 0.2280 detik pada kecepatan maksimal 5m/s menjadi 0.2451 detik pada kecepatan maksimal 10m/s dan berubah menjadi 0.3060 detik pada kecepatan maksimal 15m/s.

3. Skenario Manet yang dihasilkan oleh *node-movement*(*mobility generation*) dan dijalankan menggunakan model propagasi *TwoRayGround* dengan penambahan jumlah *node* memiliki performa sebagai berikut.
    - Performa *Packet Delivery Ratio* yang dihasilkan fluktuatif mulai dari 23.2593% pada jumlah *node* 50 *node* menjadi 33.9486% pada jumlah *node* 75 *node* dan menjadi 31.6685% pada kecepatan maksimal 15m/s.
    - Performa *Routing Overhead* yang dihasilkan memiliki nilai semakin meningkat mulai dari 115.425 paket pada jumlah *node* 50 *node* menjadi 158.940 paket pada jumlah *node* 75 *node* dan berubah menjadi 204.217 paket pada jumlah *node* 100 *node*.
    - Performa *End to End Delay* yang dihasilkan memiliki nilai *delay* yang fluktuatif mulai dari 0.2451 detik pada jumlah *node* 50 *node* menjadi 0.4281 detik pada jumlah *node* 75 *node* dan berubah menjadi 0.0878 detik pada jumlah *node* 100 *node*.
4. Skenario Manet yang dihasilkan oleh *node-movement*(*mobility generation*) dan dijalankan menggunakan model propagasi *TwoRayGround* dengan penambahan radius transmisi memiliki performa sebagai berikut.

- Performa *Packet Delivery Ratio* yang dihasilkan cenderung meningkat mulai dari 38.0785% pada radius transmisi 150m menjadi 40.5445% pada radius transmisi 200m dan menjadi 65.6490% radius transmisi 250m.
- Performa *Routing Overhead* yang dihasilkan memiliki nilai semakin meningkat mulai dari 120.446 paket pada radius transmisi 150m menjadi 132.282 paket pada radius transmisi 200m dan berubah menjadi 146.014 paket pada radius transmisi 250m.
- Performa *End to End Delay* yang dihasilkan memiliki nilai *delay* yang fluktuatif mulai dari 0.1418 detik pada radius transmisi 150m menjadi 0.0663 detik pada radius transmisi 200m dan berubah menjadi 0.0816 detik pada radius transmisi 150m.

5. Hal-hal yang mempengaruhi nilai PDR, RO, dan *End to End Delay* yang dihasilkan dari model propagasi *TwoRayGround* adalah :
    - Posisi awal *node* yang dibuat secara acak.
    - Pergerakan *node* yang dibuat secara acak.
    - Lingkungan jaringan yang digunakan.
    - Keadaan sekitar simulasi.
    - Ketinggian antenna.
    - Radius transmisi

6.2. Saran

Dalam pengerjaan Tugas Akhir ini terdapat beberapa saran untuk perbaikan serta pengembangan sistem yang telah dikerjakan sebagai berikut:

1. Dapat dilakukan dengan penambahan atau pengurangan jumlah *node* dan penambahan jumlah percobaan untuk skenario *node-movement*(*mobility generation*).

2. Dapat dilakukan modifikasi pada parameter parameter lain yang berhubungan dengan simulasi DSDV di NS2.
3. Dapat melakukan perbandingan dengan protokol routing yang lain sehingga mendapatkan hasil yang lebih baik.
4. Dapat dilakukan perbandingan dengan model propagasi yang lain agar mendapatkan nilai yang lebih baik.

DAFTAR PUSTAKA

[1] J. Macker, "Mobile ad hoc networking (MANET): Routing protocol performance issues and evaluation considerations," 1999.

[2] J. Jamal, "MANET (Mobile ad hoc network)- Characteristics and Features," eexploria, 13 November 2011. [Online]. Available: http://www.eexploria.com/manet-mobile-ad-hoc-network-characteristics-and-features/. [Accessed 26 November 2018].

[3] Afrah Daas, Khulood Mofleh, Elham Jabr, Sofian Hamad, "Comparison between AODV and DSDV Routing protocols in Mobile Ad-hoc Network," in *5th National Symposium on Information Technology: Towards New Smart World (NSITNSW)*, 2015.

[4] A. Saputra, "Pengetahuan Tentang Jaringan," 25 Maret 2009. [Online]. Available: http://de-monk.blogspot.co.id/. [Accessed 7 Juni 2018].

[5] "NS2 Simulator Official Wiki," Sourceforge, 4 November 2011. [Online]. Available: http://nsnam.sourceforge.net/wiki/index.php/Main_Page. [Accessed 30 Mei 2018].

[6] "The Network Simulator - ns-2," 20 Maret 2018. [Online]. Available: https://www.isi.edu/nsnam/ns/. [Accessed 30 Mei 2018].

[7] D. B. Close, P. H. Rubn, A. D. Robbins, R. Stallman and P. V. Oostrum, The AWK Manual, Massachusetts: Free Software Foundation, Inc., 1995.

[8] D. A. Maltz, "On-Demand Routing in Multi-hop Wireless Mobile Ad Hoc Networks," 2001.

[9] IUPUI Electrical and Computer Engineering, "Generating Traffic-Connection and Movement Files," 2005. [Online]. Available: http://www.engr.iupui.edu/~dskim/tutorials/ns2/?section=10. [Accessed 24 October 2018].

LAMPIRAN

```
1.  #
2.  # nodes: 50, pause: 10.00, max speed: 5.00, max x:
    800.00, max y: 800.00
3.  #
4.  $node_(0) set X_ 619.942699593511
5.  $node_(0) set Y_ 20.569544536636
6.  $node_(0) set Z_ 0.000000000000
7.  $node_(1) set X_ 138.638749540984
8.  $node_(1) set Y_ 110.368000100493
9.  $node_(1) set Z_ 0.000000000000
10. $node_(2) set X_ 327.044932922662
11. $node_(2) set Y_ 281.758428226633
12. $node_(2) set Z_ 0.000000000000
13. $node_(3) set X_ 661.645646957284
14. $node_(3) set Y_ 118.647274905497
15. $node_(3) set Z_ 0.000000000000
16. $node_(4) set X_ 446.442538513813
17. $node_(4) set Y_ 764.165195204867
18. $node_(4) set Z_ 0.000000000000
19. $node_(5) set X_ 478.594136319026
20. $node_(5) set Y_ 711.775361571758
21. $node_(5) set Z_ 0.000000000000
22. $node_(6) set X_ 542.241341989999
23. $node_(6) set Y_ 787.161362294472
24. $node_(6) set Z_ 0.000000000000
25. $node_(7) set X_ 413.996227949675
26. $node_(7) set Y_ 688.006042108232
27. $node_(7) set Z_ 0.000000000000
28. $node_(8) set X_ 95.759589019696
29. $node_(8) set Y_ 521.401640505423
30. $node_(8) set Z_ 0.000000000000
31. $node_(9) set X_ 543.407267385328
32. $node_(9) set Y_ 793.829409106755
33. $node_(9) set Z_ 0.000000000000
34. $node_(10) set X_ 112.824353451716
35. $node_(10) set Y_ 24.100662631201
36. $node_(10) set Z_ 0.000000000000
37. $node_(11) set X_ 206.469856143401
38. $node_(11) set Y_ 699.663545361259
39. $node_(11) set Z_ 0.000000000000
40. $node_(12) set X_ 66.142082064774
```

```
41. $node_(12) set Y_ 239.808230726074
42. $node_(12) set Z_ 0.000000000000
43. $node_(13) set X_ 421.136924530491
44. $node_(13) set Y_ 92.734128629951
45. $node_(13) set Z_ 0.000000000000
46. $node_(14) set X_ 217.341588532331
47. $node_(14) set Y_ 550.513590524631
48. $node_(14) set Z_ 0.000000000000
49. $node_(15) set X_ 157.158753622561
50. $node_(15) set Y_ 53.355463570435
51. $node_(15) set Z_ 0.000000000000
52. $node_(16) set X_ 555.528980575043
53. $node_(16) set Y_ 413.756435844428
54. $node_(16) set Z_ 0.000000000000
55. $node_(17) set X_ 167.376634761305
56. $node_(17) set Y_ 763.114789949701
57. $node_(17) set Z_ 0.000000000000
58. $node_(18) set X_ 494.367747667360
59. $node_(18) set Y_ 762.872202759008
60. $node_(18) set Z_ 0.000000000000
61. $node_(19) set X_ 276.976722295200
62. $node_(19) set Y_ 797.801082288526
63. $node_(19) set Z_ 0.000000000000
64. $node_(20) set X_ 134.669265526162
65. $node_(20) set Y_ 603.205342373511
66. $node_(20) set Z_ 0.000000000000
```

Kode Sumber 7.1 Posisi *node* dari potongan Skenario

```
1.  $god_ set-dist 0 1 3
2.  $god_ set-dist 0 2 2
3.  $god_ set-dist 0 3 1
4.  $god_ set-dist 0 4 4
5.  $god_ set-dist 0 5 4
6.  $god_ set-dist 0 6 4
7.  $god_ set-dist 0 7 4
8.  $god_ set-dist 0 8 4
9.  $god_ set-dist 0 9 4
10. $god_ set-dist 0 10 3
11. $god_ set-dist 0 11 5
12. $god_ set-dist 0 12 3
13. $god_ set-dist 0 13 1
14. $god_ set-dist 0 14 4
```

```
15.$god_ set-dist 0 15 3
16.$god_ set-dist 0 16 2
17.$god_ set-dist 0 17 5
18.$god_ set-dist 0 18 4
19.$god_ set-dist 0 19 5
20.$god_ set-dist 0 20 4
21.$god_ set-dist 0 21 2
22.$god_ set-dist 0 22 4
23.$god_ set-dist 0 23 4
24.$god_ set-dist 0 24 3
25.$god_ set-dist 0 25 3
26.$god_ set-dist 0 26 3
27.$god_ set-dist 0 27 4
28.$god_ set-dist 0 28 4
29.$god_ set-dist 0 29 3
30.$god_ set-dist 0 30 1
31.$god_ set-dist 0 31 1
32.$god_ set-dist 0 32 3
33.$god_ set-dist 0 33 1
34.$god_ set-dist 0 34 3
35.$god_ set-dist 0 35 4
36.$god_ set-dist 0 36 3
37.$god_ set-dist 0 37 2
38.$god_ set-dist 0 38 3
39.$god_ set-dist 0 39 1
40.$god_ set-dist 0 40 4
41.$god_ set-dist 0 41 2
42.$god_ set-dist 0 42 3
43.$god_ set-dist 0 43 5
44.$god_ set-dist 0 44 4
45.$god_ set-dist 0 45 3
46.$god_ set-dist 0 46 3
47.$god_ set-dist 0 47 2
48.$god_ set-dist 0 48 4
49.$god_ set-dist 0 49 4
50.$god_ set-dist 1 2 2
51.$god_ set-dist 1 3 3
52.$god_ set-dist 1 4 4
53.$god_ set-dist 1 5 4
```

```
54.$god_ set-dist 1 6 5
55.$god_ set-dist 1 7 4
56.$god_ set-dist 1 8 3
57.$god_ set-dist 1 9 5
58.$god_ set-dist 1 10 1
59.$god_ set-dist 1 11 4
60.$god_ set-dist 1 12 1
61.$god_ set-dist 1 13 2
62.$god_ set-dist 1 14 3
63.$god_ set-dist 1 15 1
64.$god_ set-dist 1 16 3
65.$god_ set-dist 1 17 4
66.$god_ set-dist 1 18 4
67.$god_ set-dist 1 19 4
68.$god_ set-dist 1 20 3
69.$god_ set-dist 1 21 1
70.$god_ set-dist 1 22 5
71.$god_ set-dist 1 23 2
72.$god_ set-dist 1 24 1
73.$god_ set-dist 1 25 4
74.$god_ set-dist 1 26 4
75.$god_ set-dist 1 27 2
76.$god_ set-dist 1 28 5
77.$god_ set-dist 1 29 4
78.$god_ set-dist 1 30 2
79.$god_ set-dist 1 31 2
80.$god_ set-dist 1 32 1
81.$god_ set-dist 1 33 2
82.$god_ set-dist 1 34 2
83.$god_ set-dist 1 35 3
84.$god_ set-dist 1 36 4
85.$god_ set-dist 1 37 3
86.$god_ set-dist 1 38 3
87.$god_ set-dist 1 39 2
88.$god_ set-dist 1 40 3
89.$god_ set-dist 1 41 1
90.$god_ set-dist 1 42 1
91.$god_ set-dist 1 43 4
92.$god_ set-dist 1 44 5
```

```
93. $god_ set-dist 1 45 1
94. $god_ set-dist 1 46 4
95. $god_ set-dist 1 47 1
96. $god_ set-dist 1 48 2
97. $god_ set-dist 1 49 4
```

Kode Sumber 7.2 Pembuatan 'GOD' dari potongan scenario

```
# nodes: 2, max conn: 1, send rate: 4.0,
seed: 1.0
#
#
# 1 connecting to 2 at time
2.5568388786897245
#
set udp_(0) [new Agent/UDP]
$ns_ attach-agent $node_(1) $udp_(0)
set null_(0) [new Agent/Null]
$ns_ attach-agent $node_(2) $null_(0)
set cbr_(0) [new Application/Traffic/CBR]
$cbr_(0) set packetSize_ 512
$cbr_(0) set interval_ 4.0
$cbr_(0) set random_ 1
$cbr_(0) set maxpkts_ 10000
$cbr_(0) attach-agent $udp_(0)
$ns_ connect $udp_(0) $null_(0)
$ns_ at 2.5568388786897245 "$cbr_(0) start"
#
#Total sources/connections: 1/1
#
```

Kode Sumber 7.3 Koneksi yang digunakan pada 'cbr.txt'

```
# A 50-node example for ad-hoc simulation
with DSDV

# Define options
# channel type
set val(chan)     Channel/WirelessChannel ;
# radio-propagation model
set val(prop)     Propagation/TwoRayGround ;
# network interface type
set val(netif)    Phy/WirelessPhy ;
# MAC type
set val(mac)    Mac/802_11 ;
# interface queue type
set val(ifq)    Queue/DropTail/PriQueue ;
# link layer type
set val(ll)     LL ;
# antenna model
set val(ant)    Antenna/OmniAntenna ;
# max packet in ifq
set val(ifqlen) 50 ;
# number of mobilenodes
set val(nn)     50 ;
# routing protocol
set val(rp)     DSDV ;
# X dimension of topography
set opt(x)      800 ;
# Y dimension of topography
set opt(y)      800 ;
# time of simulation end
set val(stop)   100 ;
set val(seed)    0 ;
set val(cp)      "cbr.txt" ;
set val(sc)      "scenario.txt" ;

Phy/WirelessPhy set RXThresh_ 1.42681e-08;
```

```
set ns_             [new Simulator]
set tracefd        [open trace.tr w]
#set windowVsTime2 [open win.tr w]
set namtrace       [open simwrls.nam w]

$ns_ trace-all $tracefd
$ns_ namtrace-all-wireless $namtrace $opt(x)
$opt(y)
# set up topography object
set topo        [new Topography]

$topo load_flatgrid $opt(x) $opt(y)

set god_ [create-god $val(nn)]
#
#  Create nn mobilenodes [$val(nn)] and
attach them to the channel.
#
# configure the nodes
        $ns_ node-config -adhocRouting
$val(rp) \
                  -llType $val(ll) \
                  -macType $val(mac) \
                  -ifqType $val(ifq) \
                  -ifqLen $val(ifqlen) \
                  -antType $val(ant) \
                  -propType $val(prop) \
                  -phyType $val(netif) \
                  -channelType $val(chan) \
                  -topoInstance $topo \
                  -agentTrace ON \
                  -routerTrace ON \
                  -macTrace OFF \
                  -movementTrace ON
```

```
      for {set i 0} {$i < $val(nn) } { incr
i } {
            set node_($i) [$ns_ node]
            $node_($i) random-motion 0;
      }
# Define node movement model
puts "Loading Conneciton Pattern ...."
source $val(cp)

#Define traffice mode
puts "Loading scenarion file...."
source $val(sc)

# Define node initial position in nam
for {set i 0} {$i < $val(nn)} { incr i } {
# 30 defines the node size for nam
$ns_ initial_node_pos $node_($i) 30
}
# Telling nodes when the simulation ends
for {set i 0} {$i < $val(nn) } { incr i } {
    $ns_ at $val(stop).0 "$node_($i) reset";
}
$ns_ at $val(stop).0002 "puts \"NS
EXITING....\";
$ns_ halt"

puts $tracefd "M 0.0 nn $val(nn) x $opt(x) y
$opt(y) rp $val(rp)"
puts $tracefd "M 0.0 sc $val(sc) cp $val(cp)
seed $val(seed)"
puts $tracefd "M 0.0 prop $val(prop) ant
$val(ant)"

puts "Starting Simulation...."
$ns_ run
```

Kode Sumber 7.4 file .tcl untuk simulasi DSDV

```
BEGIN{
      recvs = 0;
      besaran_paket = 0;
}
{
      if (($1 == "s" || $1 == "r") && $4 ==
"RTR"){
            recvs++;
      }
      if (($1 == "s" || $1 == "r") && $4 ==
"RTR")
      {
            besaran_paket = besaran_paket +
$8;
      }
}
END{
      printf("Routing Overhead : %.3f\n",
besaran_paket/recvs);
}
```

Kode Sumber 7.5 Implementasi perhitungan RO

```
BEGIN {
      sendLine = 0;
      recvLine = 0;
      forwardLine = 0;
      pdr = 0;
}
{
if ($1 == "s" && $4 == "AGT"){
      sendLine++;
}
```

```
if ($1 == "r" && $4 == "AGT"){
      recvLine++;
}

if ($1 == "f" && $4 == "RTR"){
      forwardLine++;
}
}
END{
      pdr= (recvLine/sendLine)*100;
      printf"Packet Delivery
Ratio:%.4f\n",pdr;
}
```

Kode Sumber 7.6 implementasi perhitungan PDR

```
BEGIN {
seqno = -1;
count = 0;
}
{
if($1 == "s" && $4 == "AGT" && seqno < $6) {
seqno = $6;
}
if($1 == "s" && $4 == "AGT") {
start_time[$6] = $2;
} else if(($7 == "cbr") && ($1 == "r")) {
end_time[$6] = $2;
} else if($1 == "D" && $7 == "cbr") {
end_time[$6] = -1;
}
}
END {
for(i=0; i<=seqno; i++) {
if(end_time[i] > 0) {
```

```
delay[i] = end_time[i] - start_time[i];
count++;
}
else
{
delay[i] = -1;
}
}
for(i=0; i<=seqno; i++) {
if(delay[i] > 0) {
n_to_n_delay = n_to_n_delay + delay[i]; }
}
n_to_n_delay = n_to_n_delay/count;
printf "end to end delay : %.4f Seconds\n",
n_to_n_delay; }
```

Kode Sumber 7.7 Implementasi perhitungan E2E

Installasi NS-2

Instalasi NS-2 dilakukan pada sistem operasi Ubuntu 16.04. Yang diperlukan untuk menggunakan NS-2 adalah melakukan instalasi dependensi dan *source code* ns-2.35. Sebelum melakukan instalasi NS-2 diperlukan beberapa dependensi agar NS-2 dapat dijalankan. Cara instalasi dependensi tersebut ditunjukkan pada perintah di bawah

```
sudo apt-get update
sudo apt-get install build-essential
autoconf automake libxmu-dev
sudo apt-get install gcc-4.9
```

Kode Sumber 7.8 Installasi dependensi NS-2

Setelah semua dependensi terpasang selanjutnya adalah mengunduh *source code* ns-2.35 dengan cara yang ditunjukkan pada perintah di bawah

```
wget
https://sourceforge.net/projects/nsnam/file
s/latest/download
```

Kode Sumber 7.9 Mengunduh kode sumber ns-2.35

Lalu mengekstrak file dengan kode dibawah ini

```
tar xvzf ns-allinone-2.35.tar.gz
```

Kode Sumber 7.10 Ekstrak file NS-2

Selanjutnya buka file "ns-allinone-2.35/ns-2.35/linkstate/ls.h" kemudian melakukan proses pengubahan kode yang ada pada baris ke-137 dimana tulisan *erase* yang ada pada kode dibawah ini.

```
void eraseAll() { erase(baseMap::begin(),
baseMap::end()); }
```

Kode Sumber 7.11 Line of code baris ke-137 sebelum diubah

Diubah Menjadi *this->erase* seperti kode dibawah ini.

```
void eraseAll() { this->erase(baseMap::begin(),
baseMap::end()); }
```

Kode Sumber 7.12 Line of code baris ke-137 setelah diubah

Selanjutnya buka file **"ns-allinone-2.35/otcl-1.14/Makefile.in"** kemudian ubahlah kode variabel *compiler* bahasa pemrograman C pada baris ini

```
CC= @CC@
```

Kode Sumber 7.13 Line of code varibel *compiler* C sebelum diubah

Menjadi seperti ini

```
CC=gcc-4.9
```

Kode Sumber 7.14 Line of code varibel *compiler* C setelah diubah

Kemudian langkah berikutnya, meng-install ns-2 dengan cara membuka program *terminal* dan memasukkan perintah yang ditunjukan dibawah ini.

```
cd /ns-allinone-2.35
./install
```

Kode Sumber 7.15 Perintah instalasi NS-2

Setelah proses installasi selesai, kemudian dilakukan proses men-set *environment variables* NS-2 dengan cara membuka file bashrc dengan perintah dibawah ini.

```
gedit .bashrc
```

Kode Sumber 7.16 Perintah untuk pengubahan .bashrc

Kemudian salinlah kode dibawah ini dan *paste* kode tersebut dibagian akhir. Jangan lupa mengganti tulisan "*Your-User-Name"* dengan username yang pada computer.

```
# LD_LIBRARY_PATH
OTCL_LIB=/home/Your-User-Name/ns-allinone-
2.35/otcl-1.14
NS2_LIB=/home/Your-User-Name/ns-allinone-
2.35/lib
X11_LIB=/usr/X11R6/lib
USR_LOCAL_LIB=/usr/local/lib
export
LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$OTCL_LIB:
$NS2_LIB:$X11_LIB:$USR_LOCAL_LIB
# TCL_LIBRARY
TCL_LIB=/home/Your-User-Name/ns-allinone-
2.35/tcl8.5.10/library
USR_LIB=/usr/lib
export TCL_LIBRARY=$TCL_LIB:$USR_LIB
# PATH
XGRAPH=/home/Your-User-Name/ns-allinone-
2.35/bin:/home/Your-User-Name/ns-allinone-
2.35/tcl8.5.10/unix:/home/Your-User-
Name/ns-allinone-2.35/tk8.5.10/unix
#the above two lines beginning from xgraph
and ending with unix should come on the
same line
NS=/home/Your-User-Name/ns-allinone-
2.35/ns-2.35/
NAM=/home/Your-User-Name/ns-allinone-
2.35/nam-1.15/
PATH=$PATH:$XGRAPH:$NS:$NAM
```

Kode Sumber 7.17 Proses pengubahan *line of code* .bashrc

BIODATA PENULIS

Muhammad Adnan Yusuf, lahir pada tanggal 24 September 1994 di Surakarta. Saat ini sedang menempuh perguruan tinggi di Institut Teknologi Sepuluh November Departemen Informatika Fakultas Teknologi Informasi dan Komunikasi pada tahun 2013. Terlibat aktif pada organisasi mahasiswa tingkat jurusan antara lain Departemen Hubungan Luar Himpunan Mahasiswa Teknik Computer-Informatika (HMTC) ITS sebagai staff tahun 2014 – 2015, sebagai staff Biro Keamanan dan Transportasi pada Schematics 2014, dan anggota Sie Perlengkapan dan Transportasi pada Schematics 2015 .

Apabila ingin bertanya lebih lanjut, silahkan kirim *email* ke adnanislamic45@gmail.com.